```
(NASA-CR-184686)  PROCEEDINGS OF THE                 N89-26492
WORKSHOP ON SPACE TELEROBOTICS, VOLUME 2              --THRU--
(Jet Propulsion Lab.)  422 p        CSCL 09B          N89-26540
                                                      Unclas
                                              G3/63   0188120
```

This publication was prepared by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

ABSTRACT

These proceedings report the results of a workshop on space telerobotics, which was held at the Jet Propulsion Laboratory, January 20-22, 1987. Sponsored by the NASA Office of Aeronautics and Space Technology (OAST), the workshop reflected NASA's interest in developing new telerobotics technology for automating the space systems planned for the 1990s and beyond. The workshop provided a window into NASA telerobotics research, allowing leading researchers in telerobotics to exchange ideas on manipulation, control, system architectures, artificial intelligence, and machine sensing. One of the objectives was to identify important unsolved problems of current interest. The workshop consisted of surveys, tutorials, and contributed papers of both theoretical and pratical interest. Several sessions were held with the themes of sensing and perception, control execution, operator interface, planning and reasoning, and system architecture. Discussion periods were also held in each of these major topics.

ACKNOWLEDGMENT

Thanks are due to D.R. Meldrum, R.S. Doshi, K. Kreutz, H. Seraji, M.H. Milman, and S.K. Owen for significant contributions to the organization of the workshop.

CONTENTS

Volume I

Volume III

SENSING AND PERCEPTION

S1-37
68
188121

The Sensing and Perception Subsystem of the NASA Research Telerobot

B. Wilcox, D.B. Gennery, B. Bon, and T. Litwin
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109

JJ574450

1. Abstract

A useful space telerobot for on-orbit assembly, maintenance, and repair tasks must have a sensing and perception subsystem which can provide the locations, orientations, and velocities of all relevant objects in the work environment. This function must be accomplished with sufficient speed and accuracy to permit effective grappling and manipulation. Appropriate symbolic names must be attached to each object for use by higher-level planning algorithms. Sensor data and inferences must be presented to the remote human operator in a way that is both comprehensible in ensuring safe autonomous operation and useful for direct teleoperation. Research at JPL toward these objectives is described.

2. Introduction

The JPL Robotics Laboratory has been conducting sensing and perception research since the mid 1970's, when a task was undertaken to develop a breadboard Mars rover which could navigate autonomously over unknown terrain. At that time, and continuing to the present, the principal sensor modality addressed was machine vision. This arises from the fact that it is essential, both in planetary rover and orbital tasks, to sense the environment prior to actual physical contact so that contact forces can be controlled. The available non-contact sensing techniques are limited to those based on electromagnetic radiation and those based on sound. Obviously sound is not useful in vacuum and of limited use in extremely rarified atmospheres. Electromagnetic sensing can be of an active type, emitting radiation and sensing the reflection, or passive, relying on ambient radiation. Active sensing systems can give direct information such as object range, but often consume excessive power and involve mechanical scanning devices which are potentially unreliable. Thus passive electromagnetic sensing is an attractive means of accomplishing the non-contact sensing function. The only wavelengths for which large amounts of ambient radiation exist in space are those emitted by the Sun, i.e. visible light and near IR. Sensors for these wavelengths are readily available with very good spatial and temporal resolution and accuracy in the form of solid-state video cameras. This has the further advantage that the human operator can easily comprehend the raw data from these sensors using a video display.

A useful space telerobot for on-orbit assembly, maintenance, and repair tasks must have a sensing and perception subsystem which can provide the locations, orientations, and velocities of all relevant objects in the work environment. Current goals of our research are to develop technology which will allow visual acquisition and tracking of known but unlabelled objects in space with sufficient speed and accuracy to permit effective grappling and manipulation. Examples of the potential uses of such technology are robotic systems for capturing satellites which have arbitrary and unknown motion, and robotic systems for construction in space. The vision system currently under development includes custom-designed image-processing hardware, and acquisition and tracking software running on a general purpose computer (Figure 1).

The machine vision system at JPL is designed to acquire and track polyhedral objects moving and rotating in space, using two or more cameras, programmable image-processing hardware, and a general purpose computer for high-level functions. The image-processing hardware is called PIFEX, for "Programmable Image Feature Extractor," and is capable of performing a large variety of operations on images and on image-like arrays of data. Acquisition utilizes image locations and velocities of features extracted by PIFEX to determine the 3-dimensional position, orientation, velocity and angular velocity of an object. Acquisition takes several seconds, but is adequate to initialize the object tracker. Tracking correlates edges detected in the current image with edge locations predicted from an internal model of the object and its motion, continually updating velocity information to predict where edges should appear in future frames. Once tracking has begun, it processes some 10 frames per second, thus allowing real-time tracking of objects.

3. PIFEX

PIFEX is a pipelined-image processor being built in the JPL Robotics Lab. It is a programmable system that will perform elaborate computations whose exact nature is not fixed in the hardware, and that can handle multiple images. It thus is more versatile than previous pipelined-image processors. It also is a very powerful system. A moderate-sized PIFEX costing less than \$100,000 will be able to perform about 10^{10} 12-bit elementary operations per second. PIFEX is a powerful, flexible tool for image processing and low-level computer vision. It also has applications in other two-dimensional problems such as route planning for obstacle avoidance and the numerical solution of two-dimensional partial differential equations.

PIFEX contains three types of programmable operators (Figure 2): convolvers, neighborhood comparison operators, and binary functions. The convolvers use a 3-by-3 kernel. Larger kernels can be simulated through the use of multiple convolvers, although this is efficient only in special cases. The neighborhood comparison operators produce a nonlinear function of the pixels in a 3-by-3 neighborhood. They are useful for such things as finding peaks, ridges, valleys, and zero crossings, as well as for region growing, shrinking, and other cellular operations. The binary functions receive two inputs and compute any desired function of their corresponding pixel values, by means of table lookup with linear interpolation. PIFEX consists of an array of identical modules, each of which contains two convolvers, one binary function, and one neighborhood comparison operator.

The modules are connected in a regular pattern in which each of two outputs from each module branches to the inputs of several

PAGE 2 INTENTIONALLY BLANK

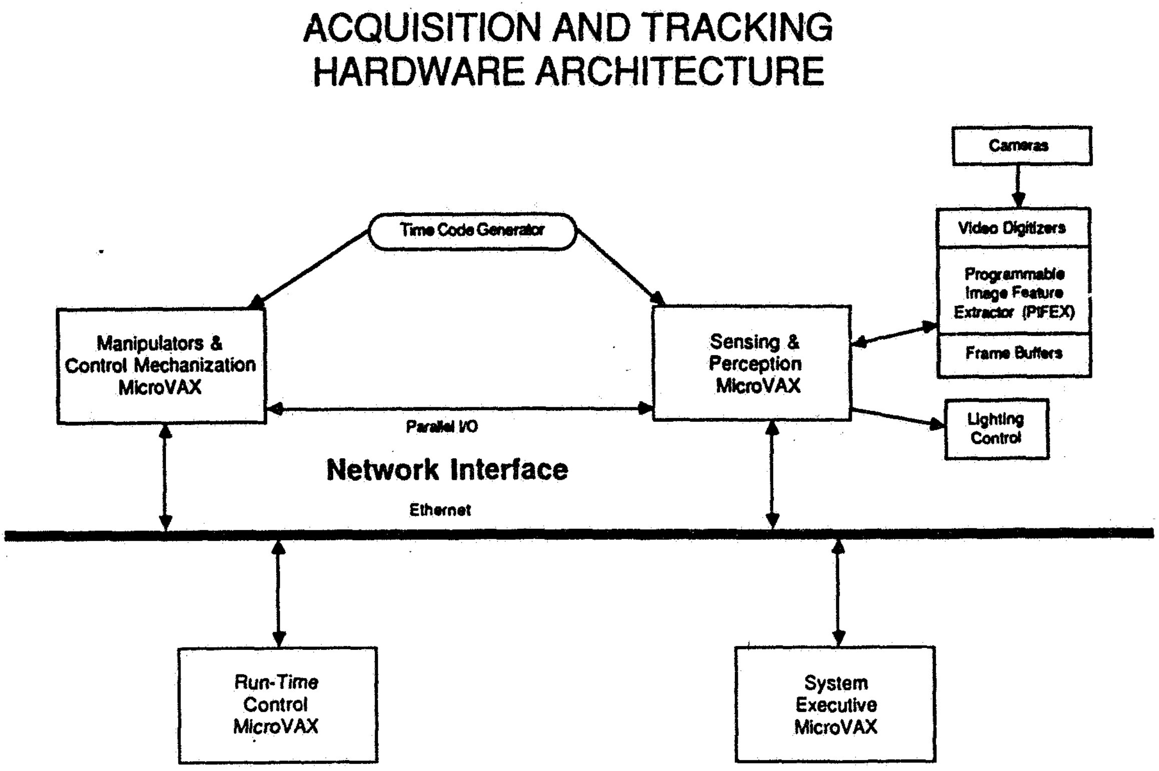

ACQUISITION AND TRACKING HARDWARE ARCHITECTURE
Cameras
Video Digitizers
Programmable Image Feature Extractor (PIFEX)
Frame Buffers
Time Code Generator
Manipulators & Control Mechanization MicroVAX
Sensing & Perception MicroVAX
Lighting Control
Parallel I/O
Network Interface
Ethernet
Run-Time Control MicroVAX
System Executive MicroVAX

Figure 1. Acquisition and Tracking Hardware Architecture

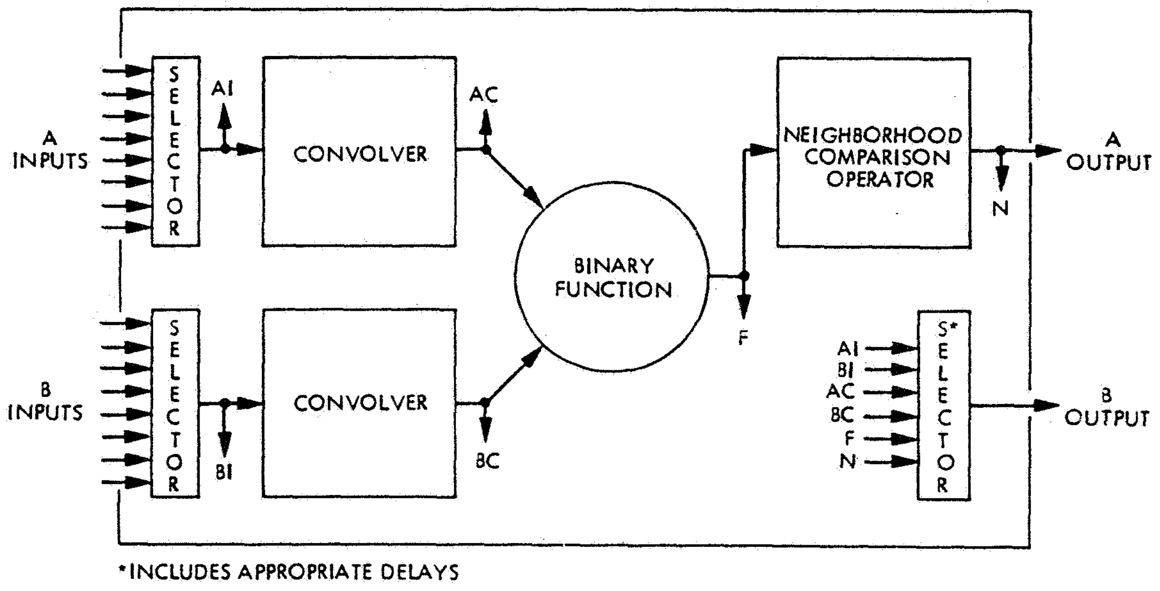


Figure 2. PIFEX module

different modules. The outputs from the modules in each column in the pattern are connected to the inputs of modules in the next column, so that the main data flow is considered to be from left to right. In this way, synchronism is achieved, since all of the modules in a given column (except for the wrap-around of rows discussed below) are processing corresponding pixels at the same time. Different rows of modules correspond to parallel data paths, but these different paths can communicate with each other because of the branching of the connections from one column to the next.

Each row is considered to wrap around to form a loop. The fanout pattern continues cyclically around these loops, except that after one particular column there are switches that can break each connection between the output of a module and the fanout to the next column, so that outputs can be extracted here and inputs can be inserted. This row wrap-around feature is important for efficient coding of algorithms that vary greatly in the width and length of data paths that they require, since an algorithm that requires a long path can wrap around several times, using only as many parallel data paths at any point as it needs. The upward bias of the fanout helps in spiraling the paths upwards in order to avoid collisions. (Since the pixels have been delayed by different amounts on different times around, ordinarily data from these different paths should not be combined with each other.) Also, each column wraps around to form a loop, and the fanout pattern is cycled invariantly around the loops. This feature is convenient for algorithms that just barely fit, since crossing the boundary that otherwise would exist at the top and bottom may help in making the necessary connections. In particular, the column wrap-around feature makes it practical to extract the outputs at the same rows at which the inputs are inserted in cases where row wrap-around is used, so that modules are not wasted.

The two wrap-around features combined cause the the interconnections of the modules in PIFEX to have the topology of a torus. There is a cut around the torus at one place to allow inputs (from image buffers or TV cameras) and outputs (to image buffers) to be switched in (as stated above), under control of the host computer.

It is planned that the initial version of PIFEX will have 5 columns and about 24 rows. (Thus it would be possible to code algorithms that vary from requiring a data path 24 modules wide and 5 modules long to requiring a data path one module wide and 120 modules long, without having to use separate passes through PIFEX on separate frame times.)

Even though the chosen approach results in a physically larger device (and perhaps greater cost if produced in quantity) than other possible approaches, it has the advantages of quicker and less expensive development (because of the need for fewer types of complicated custom VLSI chips), ease of computing arbitrary functions (because of the generality of the table-lookup functions), and easy growth to a more powerful system (because of the modular concept with the regular interconnection pattern).

PIFEX has been described in more detail elsewhere.[1,2]

4. Acquisition and Tracking

The organization of the acquisition and tracking system is shown in Figure 3. Its operation will be described briefly in this section.

The Feature Tracker detects features in the images from each camera, tracks them as they move over time, smooths their two-dimensional positions, and differentiates the positions to obtain their two-dimensional velocities in the image plane. (The features currently used correspond to the vertices of a polyhedral object.) Features that are not moving, are moving too fast, or do not remain sufficiently long are rejected. Future versions of the Feature Tracker may also measure other properties of the features in addition to position, such as orientation, to aid in stereo matching and in matching to the object model. The Feature Tracker will run primarily in PIFEX when it becomes available.

When enough features are being tracked, the Motion Stereo module uses the information from all of the cameras for some particular time to compute the partial three-dimensional information. For a single camera, the object range is completely indeterminate, but the relative ranges of the features are determined using the assumption that they are connected to a rigid, moving object. For multiple cameras, the absolute ranges of the features in general can be determined. This includes the three-dimensional position of each feature (from any camera), an estimate of its position accuracy as given by a 3-by-3 covariance matrix, and estimates of the velocity and angular velocity of the object. All of this information is based on nominal values of unity for scale factor and zero for bias. In addition, a 2-by-2 covariance matrix of the uncertainty in these nominal values of scale factor and bias is estimated. The motion stereo algorithm has been described in more detail elsewhere.[3]

The Stereo Matcher refines this information and computes estimates of the scale factor and bias. It uses a general matching process based on a probabilistic search.[4] In this process, features from one camera are matched one at a time to features from another camera in order to build a search tree. For each combination of trial matches, a least-squares adjustment is done for the scale factor and bias that produces the best agreement of the matched features. The discrepancies in the adjusted positions of the matched features compared to their accuracies are used to compute a probability for each match combination, and these probabilities are used to prune the search. If there are more than two cameras, the current plan is for the Stereo Matcher to use only a specified pair for matching, but more elaborate arrangements are possible.

The Model Matcher matches the three-dimensional feature positions (and any other feature information available) to those of the object model in order to determine the three-dimensional position and orientation of the object, by using a search process similar to that in the Stereo Matcher. This information is valid for the time at which the features had these positions. However, time has elapsed since then, while the computations in the Motion Stereo module, Stereo Matcher, and Model Matcher were being done.

Meanwhile, the Feature Tracker, running concurrently with the other modules, still has been tracking the features (those that have remained visible). The latest positions of these features, together with the information from the model matcher that indicates which object features they match, are used by the Tracking Initializer to update the object position and orientation to the time of this most recent data. Also, the two-dimensional velocities from the Feature Tracker are used to compute the three-dimensional velocity and angular velocity of the object for this time. The solution for position and orientation in the Tracking Initializer is an iterative nonlinear weighted least-squares adjustment. The initial approximation for this iterative solution is obtained from the old position and orientation from the Model Matcher, extrapolated to the new time by using the velocity and angular velocity from the Motion Stereo module, as corrected by using the scale factor and bias estimates from the Stereo Matcher. The accuracy estimates of the solution, in the form of a 12-by-12 covariance matrix of position, orientation, velocity, and angular velocity, also are produced.

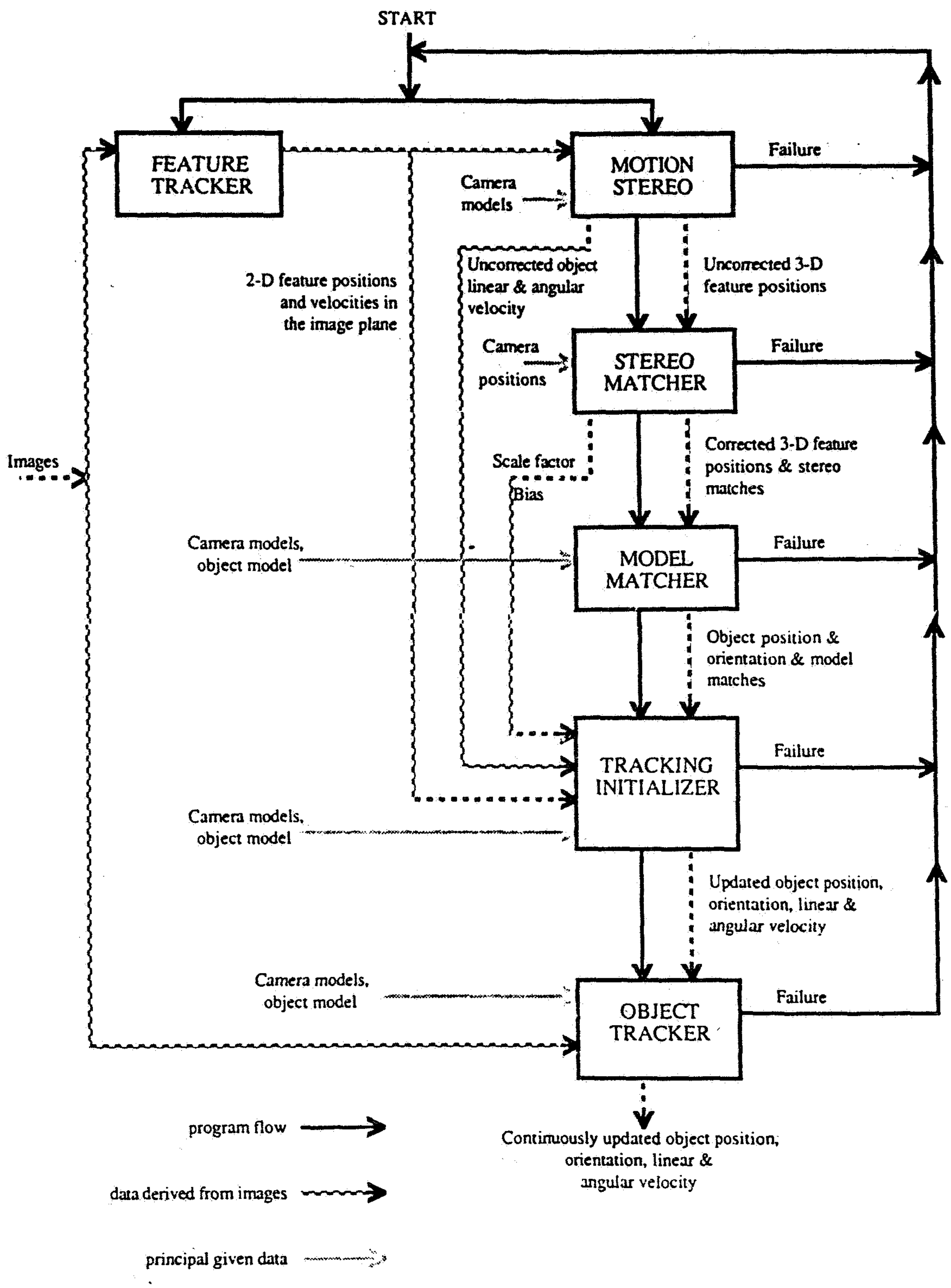

START
FEATURE TRACKER
MOTION STEREO
Camera models
Failure
2-D feature positions and velocities in the image plane
Uncorrected object linear & angular velocity
Uncorrected 3-D feature positions
Camera positions
STEREO MATCHER
Failure
Images
Scale factor
Bias
Corrected 3-D feature positions & stereo matches
Camera models, object model
MODEL MATCHER
Failure
Object position & orientation & model matches
TRACKING INITIALIZER
Failure
Camera models, object model
Updated object position, orientation, linear & angular velocity
Camera models, object model
OBJECT TRACKER
Failure
Continuously updated object position, orientation, linear & angular velocity
program flow
data derived from images
principal given data

Figure 3. Acquisition and Tracking Software Architecture.

The position, orientation, velocity, angular velocity, and their covariance matrix from the Tracking Initializer are used as initial conditions in the Object Tracker. It rapidly and accurately updates this information. Currently, the features that it looks for in the images are the object edges. Using edges produces more complete information than using vertices. Edges can be used easily here, because the one-dimensional information from edge elements suffices once the approximate object position and orientation are known. The edges currently are detected by IMFEX,[5] which is a nonprogrammable precursor to PIFEX and computes an approximation to the Sobel operator. When PIFEX is available, it will detect the edges and perform a portion of the computation involving them.

The object tracker works in a loop with the following major steps: prediction of the object position and orientation for the time at which a picture is taken by extrapolating from the previously adjusted data (or from acquisition data when starting); detection of features by projecting into the picture to find the actual features and to measure their image positions relative to the predictions; and the use of the resulting data to adjust the position, orientation, and their time derivatives so that the best estimates for the time of the picture are obtained. The object tracker has been described elsewhere.[6]

It is possible for any of these modules to fail because of poor data. A failure in any of them causes the acquisition process to start over. As new features become visible, they may contain good enough information for successful acquisition and tracking. (The Object Tracker can track through regions of data too poor to allow acquisition to occur. If it fails, the re-acquisition probably will not succeed immediately, but eventually the object may move into a region of sufficient visibility for acquisition.)

Notice that in the Model Matcher, the Tracking Initializer, and the Object Tracker stereo information is used implicitly. That is, stereo matching between cameras need not be done for all features used by the Model Matcher and the Tracking Initializer, and it is not done for any features in the Object Tracker. Instead, features are matched directly to the object model. (In all three modules, these features can come separately from each camera, but in the Model Matcher and Tracking Initializer, those features that have been matched between cameras by the Stereo Matcher are used as units.) This process produces accurate stereo depth information even if the same features are not seen by different cameras, because of the rigid-body constraint in the object model.

This approach can be extended directly to multiple object recognition. Since one of the outputs of the model matcher is a probability of correct match, several of these matchers working in parallel with different object models could perform the recognition function. However, if a large number of different objects need to be recognized, additional modules would need to be created to classify the feature patterns into one of several groups before an attempt to make a detailed match. These broad classifications of objects might be made on the basis of the presence of cylindrical or spherical surfaces or the number of features of a given type (edge, vertex, etc.).

5. Camera calibration

The grappling of a spinning or tumbling satellite requires that the manipulator control system and the machine vision system agree on the 3-D positions of objects in the work volume. To achieve this correspondence, a calibration fixture has been fabricated that is used for both manipulator calibration[7] and camera calibration. This fixture has an array of dots machined on a black-anodized aluminium plate, mounted on a framework which can be affixed to the floor of the research facility in any one of nine pre-measured positions. These positions include three different planes for the face of the plate, so that the dots on the array are seen by the cameras at three different distances, allowing accurate determination of the camera parameters.

The first step in camera calibration is to capture images from the various cameras of the calibration fixture in each of the measured positions. Manual input consists of the following; the camera number, the position number of the measured fixture position, the 3-D coordinates of these positions, the spacing of the dots, the diameter of the dots, the number of rows and columns of dots, the nominal focal length of the camera, the nominal pixel spacing, and the approximate 3-D position of the camera. At present, the operator designates the corner dots. The result is a set of points, with for each point its 3-D position and its measured 2-D position.

First, the approximate dot spacing a (in pixels) in the image plane is computed from the designated corner dot positions. Then the approximate Gaussian function for filtering is defined so that its standard deviation is half of the average dot spacing. The image is low-pass filtered by convolving with a one-dimensional Gaussian function first along the columns and then along the rows, and the result is subtracted from the original image to obtain the high-pass-filtered image.

A histogram of the high-pass-filtered image b_h is computed for the portion of the image which is expected to include all but the outer rows and columns of dots, with buckets for every integer from -255 to 255. This is summed and normalized to produce the cumulative distribution c_h. The predicted portion of area covered by the dots is computed from the known size and spacing of the dots. Then values halfway between this and 0 and 1 are computed, and the brightness values for which the cumulative histogram is equal to these values are found. The average of these two brightness values is used as the threshold.

Every pixel of the high-pass-filtered image within the expected area of the dots whose value exceeds that of the threshold is tentatively assumed to be part of a dot. Every connected (by four-neighbor connection) area of such pixels is examined to see if it forms a good dot. Its area should be within four pixels plus 10% of the expected value, and the Euclidean distance of its border pixels from the centroid of its pixel positions should not vary by more than one pixel plus 5%. The dots that pass these tests are used, and the others are rejected. For each dot that passes, the centroid of its pixel positions is used as the 2-D dot position (to sub-pixel accuracy). The 3-D dot position is obtained from the known dot spacings, with the individual dots being identified by progressing one dot at a time from a known corner dot according to the expected dot image spacing.

6. Camera Model Adjustment

The actual camera model adjustment is performed by a least-squares adjustment, which finds the set of camera model parameters that minimizes the sum of the squares of the differences btween the predicted postions and measured postitions (in two dimensions) of the dots on the calibration fixture. The form of the camera model is that described in Yakimovsky and Cummingham 1978[8], although we will probably add two terms for lens distortion to the model later. The least-squares adjustment is performed iteratively, since the problem is nonlinear. Also, in case the dot finder makes mistakes, automatic editing is done to remove bad dots, using the method described in Gennery 1980 and Gennery 1986.[9,3]

7. Status

A wire-wrapped prototype PIFEX module has been produced and debugged, using a version of the convolver composed of three custom VLSI chips (plus the line buffers). A printed circuit layout is being designed for use with a single-chip convolver, leading to production of a PIFEX with about 120 modules. A high-level language for programming PIFEX has been designed, and a compiler will be written for it.

The acquisition and tracking system has been designed, and most of it has been coded in Pascal for the microVAX-II. The Feature Tracker, Motion Stereo module and Stereo Matcher have executed successfully. The Model Matcher is still under development, and coding has begun on the Tracking Initializer. The Object Tracker was running on a different computer from the VAX presently in use; it has been translated for use on the VAX but has yet to run on real images there. Once all modules are working, optimization and integration will begin. Finally, when a sufficiently large PIFEX is available, appropriate parts of acquisition and tracking, including much of the Feature Tracker, will be programmed into PIFEX, thus increasing the speed and robustness of the system.

8. Acknowledgments

The research described in this publication was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

9. References

[1] D. B. Gennery and B. Wilcox, "PIFEX: An Advanced Programmable Pipelined-Image Processor," JPL Publication 84-97, Jet Propulsion Laboratory, Pasadena, CA, 1984.

[2] D. B. Gennery and B. Wilcox, "A Pipelined Processor for Low-Level Vision," *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, June 1985, pp. 608-613.

[3] D. B. Gennery, "Stereo Vision for the Acquisition and Tracking of Moving Three-Dimensional Objects," in *Techniques for 3-D Machine Perception* (A. Rosenfeld, ed.), North-Holland, 1986.

[4] D. B. Gennery, "A Feature-Based Scene Matcher," *Proceedings of the Seventh International Joint Conference on Artificial Intelligence,* Vancouver, August 1981, pp. 667-673.

[5] R. Eskenazi and J. M. Wilf, "Low-Level Processing for Real-time Image Analysis," JPL Publication 79-79, Jet Propulsion Laboratory, Pasadena, CA, 1979.

[6] D. B. Gennery, "Tracking Known Three-Dimensional Objects," *Proceedings of the AAAI Second National Conferencc on Artificial Intelligence,* Pittsburgh PA, August 1982, pp. 13-17.

[7] S. Hayati and M. Mirmirani, "Improving the absolute positioning accuracy of robot manipulators", *Journal of robotic systems,* Vol II, no 4, 1985.

[8] Y. Yakimovsky and R. T. Cunningham, "A System for Extracting Three-Dimensional Measurements from a Stereo Pair of TV Cameras," *Computer Graphics and Image Processing* 7, pp. 195-210 (1978).

[9] D. B. Gennery, "Modelling the Environment of an Exploring Vehicle by Means of Stereo Vision," AIM-339 (STAN-CS-80-805), Stanford University, Computer Science Dept, June 1980.

S2-63
108.
188122

Knowledge-Based Vision for Space Station Object Motion Detection, Recognition, and Tracking

P. Symosek, D. Panda, S. Yalamanchili, and W. Wehner III
Honeywell Systems and Research Center
Minneapolis, MN 55418

HY 989092

Abstract

Computer vision, especially color image analysis and understanding, ~~is~~ has much to offer in the area of the automation of Space Station tasks such as construction, satellite servicing, rendezvous and proximity operations, inspection, experiment monitoring, data management and training. Knowledge-based techniques improve the performance of vision algorithms for unstructured environments because of their ability to deal with imprecise a priori information or inaccurately estimated feature data and still produce useful results. Conventional techniques using statistical and purely model-based approaches lack flexibility in dealing with the variabilities anticipated in the unstructured viewing environment of space.

Algorithms developed under NASA sponsorship for Space Station applications to demonstrate the value of a hypothesized architecture for a Video Image Processor (VIP) are presented. Approaches to the enhancement of the performance of these algorithms with knowledge-based techniques and the potential for deployment of highly-parallel multi-processor systems for these algorithms are discussed.

1.0 Introduction

A major consideration in the design and deployment of the NASA Space Station is the definition of automation techniques which will guarantee the timely and reliable performance of the Space Station's missions. During specification of the initial design of the Space Station, NASA has identified three criteria for the justification of the development of an automation technique:1

1. The automation capability should be of substantial value toward the objective of accomplishing Space Station functions, such as user experiment monitoring, user production activities and satellite servicing in a timely and reliable manner.

2. The safety of the crew must not be compromised.

3. The Space Station should operate autonomously with as little support from ground-based facilities as possible.

A Video Image Processor will be a very valuable automation tool on-board the Space Station for several reasons: Image processing, specifically the identification of the objects seen in the image and the formulation of a 3-dimensional model of a scene, is a pre-requisite capability for the development of autonomous robots. These autonomous robots could perform many of the mundane tasks such as experiment monitoring and proximity operations that at the present time require crew member supervision. Image processing is also a pre-requisite capability for the task of bandwidth reduction which will be necessary for the Space Station because of limited on-board storage and the restraints of secure channel downlinks from the Space Station to ground-based facilities. For semi-autonomous implementations, image processing is employed to execute repetitive tasks such as color image enhancement/restoration or operator cueing, and an operator is required only for verification confirmation of the actions of the algorithms. A Video Image Processor can perform each of the preceding tasks, thus increasing the efficiency of crew members of the Space Station.

A Video Image Processor is a dedicated processing unit for image data that is modularly extendable and is to be built from commercially available components. The VIP architecture was defined conceptually under contract to NASA by Honeywell Systems and Research Center on the basis of several criteria, which are: maintainability, extensibility, programmability, physical aspects of deployment and the performance specifications defined by current Space Station applications2. The candidate architectures for the VIP were quantitatively evaluated with architecture analysis tools to obtain a high degree of confidence in achieving the desired functionality. To do this a set of example image processing algorithms had to be specified and their performance evaluated for imagery acquired during previous Space Shuttle missions to simulate the algorithms' behavior under realistic conditions. In this way, the processing requirements of the algorithms could be estimated for the unique set of environmental, lighting and imaging constraints found in space.

The goal of the selection process was to develop an algorithm suite that would benefit a sufficiently large number of space station tasks. The various space station tasks that benefit from an image processing capability can be classified into eight generic categories:3

- Construction
- Satellite servicing
- Rendezvous and proximity operations
- Inspection
- Payload delivery and retrieval
- Experiment monitoring
- Data management and communication
- Training

In order for the VIP to assist in the automation of these tasks, it should have a substantial array of image processing algorithms that it can apply in accordance with the changing demands of the application. These image processing algorithms can be grouped into six major families for the space station scenario:

- Color image enhancement
- Tracking
- Surveillance
- Identification

- Proximity operations
- Bandwidth reduction

These six families do not represent the entire breadth of the state of the art in image processing, but most of the image processing algorithms required for the automation of space station tasks belong to one of these families. In addition, algorithms in each of these categories are sufficiently mature for the design and build of a prototype system.

A cross-reference of Space Station algorithmic functions and Space Station tasks is presented in Table 1. There are several important observations to be made from Table 1. Most important, it is apparent that color image enhancement is required for all of the selected Space Station tasks.

This can be attributed to two factors. First, imaging systems are not perfect and provide noise-degraded images even under the best conditions. Imaging systems that employ electron-beam scanning mechanisms for transforming the optical image into an analog signal always exhibit random noise in their outputs. This noise is caused by thermal and electrical noise in the imaging system hardware. When an analog signal is transformed into a digital array of image intensities, there is additional noise superimposed onto the image because of the finite response time of the digitization amplifiers.

Even if the image's quality is very good, which usually will be the case if the imaging system is built with charge-coupled devices (CCD), the performance of the image processing algorithms can be improved by attenuating the remaining noise with color image enhancement algorithms. The reason that noise may remain even for a good imaging system is that the images being used are real-world images. Objects in the real world will always exhibit small perturbations in reflectivity because of variations in surface smoothness, and therefore images of these objects will always appear to be noisy. Color image enhancement algorithms can sufficiently attenuate the unwanted noise information so that its magnitude will be below the detection thresholds of the image processing algorithms.

Generic Tracking algorithms are also a pre-requisite capability for all of the Space Station tasks considered. This can be explained as follows: Scene interpretation algorithms can be decomposed into four stages:

1. Segmentation
2. Feature Detection
3. Iconic (Pixel-Based)-to-Symbolic Feature Mapping
4. Classification

The algorithms that together form the Tracking function are equivalent to the first three stages described above. Therefore, an evaluation of the characteristics of an architecture for the Tracking functions defines a dependable measure of the performance of the architecture for most high-level image processing applications. Also, the partitioning of the image into regions of interest and extraneous background is the initial phase of all image processing algorithms that are designed to obtain symbolic information from raw image data.

Of the remaining algorithm categories, Bandwidth Reduction was chosen for verification because 1). It is a pre-requisite for five of the eight Space Station tasks considered and 2) The algorithms that perform the Bandwidth Reduction function are exactly those that are required for the Tracking function, with the exception the temporal silhouette matching algorithm required for Tracking.

Knowledge-based techniques are a means of employing the efficient symbolic pattern matching and high-level reasoning capabilities of artificial intelligence for image interpretation applications. Knowledge-based techniques for region labeling can tolerate large errors in feature data and still produce meaningful results. They can be designed in stages because of their modular rule database: as new contexts are discovered for classification of features, the system is reconfigured by the definition or modification of a few rules. Knowledge-based systems are very efficient for the task of performing retrieval operations on large symbolic databases on the basis of relational and contextual constraints on the data.

Due to the unpredictable nature of imagery obtained in space, especially during construction of the Space Station itself with remotely guided robots, and other factors unique to a space environment such as rapid diurnal changes while a vehicle is in orbit, knowledge-based techniques will play a very important role in improving image interpretation algorithm performance. Knowledge-based techniques have been applied extensively for all of the four stages of image interpretation algorithms. These systems have been used for photointerpretation applications,[4-6] autonomous weapon delivery systems,[7] and the labeling of features in arbitrary urban scenes.[8-9] These expert systems have several features in common: A database of calculated image features is matched with predicates of production rules, which are represented as logical statements of the form "If ..., then..." and a control system that supervises rule activation.

The system developed by Nagao and Matsuyama[5] uses a knowledge base representing relational, contextual and geometric constraints for the task of region labeling for multi-spectral imagery obtained from low-flying aircraft. The region boundaries are detected by a variety of low-level image segmentation algorithms and the resultant information is archived on a blackboard shared by each of the experts of the system. Each expert is optimized for locating a specific kind of object or region. They devised an approach for the reliable classification of vegetational regions that is independent of the time of year, using the ratio of two distinct spectral bands to discriminate the vegetation regions from the non-vegetation regions. They demonstrate that knowledge-based techniques permit the reliable identification of houses and roads in congested urban scenes where other classical approaches normally fail.

Ohta[9] developed a hierarchical region labeling scheme for color images of urban scenes. The approach is hierarchical because an initial plan image is derived and labeled before a more detailed, data-directed segmentation is carried out. The plan image is defined by a region-based color image segmentation algorithm. The macro-level regions of the plan image are: sky, tree, building and road. These region categories are detected using top-down contextual and spectral constraints. The algorithm is very reliable and can correctly label regions in urban outdoor scenes using only 57 rules.

However, very little work has been done in the area of the application of knowledge-based techniques for tracking or motion understanding. This is primarily due to the unconstrained nature of the problem. It is exceedingly difficult to specify an expert system that can characterize the dynamic behavior of arbitrary objects as they translate and rotate in three dimensions. Work has been done on dynamic environment understanding for a mobile robot employing an intelligent system for reasoning about the three-dimensional structure of a stationary environment[10]. This approach uses path planning techniques for the identification and avoidance of obstacles detected by sonar sensors. The technique was successfully demonstrated for the task of obstacle avoidance while navigating an M113 autonomous vehicle, built by FMC Corporation for the DARPA ALV program, through a maze of obstacles. However, no procedure is specified for the detection and interpretation of sensor data that results from moving objects.

Honeywell is carrying out research on the utilization of knowledge-based techniques for the discrimination of moving objects from stationary objects in imagery obtained from a moving autonomous vehicle's camera.[11] This approach, which we have labeled Dynamic Reasoning from Integrated Visual Evidence (DRIVE), identifies visual cues from a sequence of images that defines a global dynamic reference model. Object recognition, world knowledge and the accumulation of evidence are used to disambiguate the situation and refine the global reference model.

The design principles for the identification of the optimal architecture for the Video Image Processor were reviewed in this section. The figures of merit for the design were stated and the approach utilized to validate the design for space imagery were summarized. The rationale for selecting the three image processing algorithms which were studied was explained. The next section discusses the technical details of the three image processing algorithms and specifies approaches for executing the same algorithms with knowledge-based techniques. Section 3 summarizes approaches that may be pursued for the development of image interpretation systems that employ knowledge-based techniques. The fourth section describes experimental results obtained for the algorithm validation task for the Video Image Processor. Section 5 discusses a few important problems that are yet to be solved and that can produce significant increases in algorithm efficiency and reliability for image processing in a space scenario.

2.0 VIP Algorithms

The details of the implementations of the three image processing functions chosen for the establishment of space processing requirements for the VIP are discussed in this section. Special attention is paid to citing how the performance of each algorithm is enhanced with knowledge-based techniques.

2.1 Color Image Enhancement--The following is a description of three of the algorithms that were evaluated for Color Image Enhancement. Each algorithm is designed to restore a particular feature of the color images, e.g., dynamic range, sharpness, etc. It is therefore conceivable that an actual implementation may use combinations of these algorithms to produce imagery with specific color characteristics.

2.1.1 Color Image Balanced Histogram Equalization--Color image balanced histogram equalization enhances image contrast and increases image dynamic range. The algorithm operates with the same fundamental principle that monochrome image histogram equalization employs, namely, that the gray levels of the original image are redistributed so that the histogram of the transformed image will take the form of a uniform distribution across a specified range of gray levels. This range is usually the display range of the display device. The mapping is one-to-one; thus, for each gray level of the original image, every pixel that appeared with that gray level will appear with a unique gray level in the transformed image. However, multiple gray levels from the original image can map to a single gray level in the transformed image.

For color images, histogram equalization is not a computationally simple process because of the requirement that the hue of each region remains the same before and after histogram equalization. To meet this constraint, the color image balanced histogram equalization algorithm calculates the equalization mapping for the intensity image, where the intensity image is obtained as the average of three primary images. The transformed primary images are calculated from the histogram-equalized intensity image. In this manner, the hue of each region of the image remains constant. The algorithm operates as follows. The offsets of the color image intensities from the average intensity level are calculated. and the transformed color levels are calculated as the transformed intensity level plus the original offsets. For example, consider a single pixel. If the three original images' intensity values were red = 140, green = 150, and blue = 110, then the average intensity at that point is I = 133. If the mapping derived by histogram equalization was 133→175, then the output color levels for that location are red = 182, green = 192, and blue = 152.

Images transformed with this algorithm will exhibit full dynamic range, and the hues of the regions of the image will not change. This may be shown as follows. A three-channel color image can be equivalently represented by an HIS image, where HIS stands for hue-intensity-saturation. The hue image represents the color of the regions of the original color image, where the magnitude of the hue is proportional to the percentages of the three primary colors, red, green, and blue. The intensity image is simply the arithmetic average of the three color images. The saturation image represents the strength of the color. The range of the hue image is 0 to 359, the range of the intensity image is 0 to 255, and the range of the saturation image is 0 to 1. (The hue and intensity images can be archived as integer arrays, but because the range of the saturation image is 0 to 1, it is archived as a real-valued array.)

For the HIS color space, the hue is calculated as a function of the ratio of linear functions of the three color image intensities. Specifically, this function is

$$\text{Hue} = \cos^{-1}\left[\frac{\frac{1}{2}\{(R-G)+(R-B)\}}{\sqrt{(R-G)^2+(R-B)(G-B)}}\right]$$

where R, G, and B are the red, green, and blue intensities. If B>G, then the hue = 2π-hue.
When R = G = B, the hue is undefined.

Let the three original chromatic levels at each pixel be represented as $R = I + \Delta_R$, $G = I + \Delta_G$, and $B = I + \Delta_B$, where I is the intensity value of the pixel in the original image. Let the output color levels be $R' = I' + \Delta_R$, $G' = I' + \Delta_G$, and $B' = I' + \Delta_B$, where I' is the intensity value of the pixel after transformation. Because R-G = R'-G', R-B = R'-B', and G-B = G'-B', the magnitude of the hue is unchanged by the transformation. Therefore, the color information that was present in the original scene, but was not discernable because of the low dynamic range of the image, is preserved. This characteristic of the algorithm will guarantee that the transformed image is a good representation of the original scene because its colors are faithfully reproduced.

A knowledge-based approach for adaptive Color Image Balanced Histogram Equalization would use a measure of an image's contrast in a local region to determine whether the dynamic range was low (perhaps caused by shadowing) and apply the algorithm to that region. The algorithm could employ information obtained from previous frames' processing results to assist in the identification of shadows.

2.1.2 Color Image Accentuation--Color image accentuation is a process whereby the image's sharpness is augmented by increasing the saturation of the image. Color saturation may be increased as follows. The offsets of each of the original image intensities from the intensity image values are calculated, and each of these offsets is amplified by a factor K, where $K > 1$.

We can represent the quantities MAX and MIN (where MAX and MIN are the maximum and minimum of the three primary image intensities):

$$MAX = I + \Delta_{MAX}$$
$$MIN = I + \Delta_{MIN}$$

The magnitudes of MAX and MIN, after transformation, are defined as

$$MAX' = I + K\,\Delta_{MAX}$$
$$MIN' = I - K\,\Delta_{MIN}$$

The transformed image's saturation is

$$S' = \frac{MAX' - MIN'}{MAX'}$$

$$= \frac{I + K\,\Delta_{MAX} \; - I + K\,\Delta_{MIN}}{I + K\,\Delta_{MAX}}$$

$$= \frac{\Delta_{MAX} + \Delta_{MIN}}{\frac{I}{K} + \Delta_{MAX}}$$

As $K \rightarrow \infty$, the term I/K in the denominator will decrease, thereby causing the saturation to increase. With the contraints that Δ_{MAX} is positive, Δ_{MIN} is positive, $K\Delta_{MIN} \leq I$, and $K\Delta_{MAX} \leq 255 - I$, the maximum value that S' can attain is 1, as $K \rightarrow \infty$. It can be seen from the following arguments that the hue of each region is unchanged. The color levels of the image after accentuation can be represented as $R' = I + K\Delta_R$, $G' = I + K\Delta_G$, and $B' = I + K\Delta_B$. The magnitudes of each of the subtracted pairs--R' - B', R' - G', and G' - B'--are equal to K times the magnitude of the respective subtracted pair before accentuation. When these values are used to calculate the hue for a specific pixel of the image, the factor K can be brought out of the numerator and the denominator, which means that the magnitude of the hue component will not change. Therefore, this algorithm also faithfully represents the hue of the original image.

This technique may be employed to increase the saturation of colors of each of the regions of the image. The effect is to make small surface detail more distinguishable.

2.1.3 Constrained Inverse Filtering--Constrained inverse filtering is a technique whereby degradations of the imaging process, such as a dispersing medium between the imaging system and the object of interest or out-of-focus optics, are corrected with digital signal processing. Constrained inverse filtering is effective when the point spread function (PSF) of the distorting medium or the imaging system optics is known or can be estimated fairly accurately.

Constrained inverse filtering is a specific form of inverse filtering. It is a restoration technique that attempts to invert the effects of an optical transfer function on an image. Inverse filters are implemented to minimize the sum of squared errors between the original image and the restored image for a specific model of the image formation process. Constrained inverse filters attempt to minimize the same error function, with a constraint that the norm squared of the restored image is as small as possible. This constraint is applied to prevent the noise present in the observed image from appearing at too great a level in the restored image. The model of the image formation process employed is

$$g(t, w) = h(t, w) * f(t, w) + n(t, w)$$

where g(t, w) is the observed image, h(t,w) is the PSF of the degradation function, f(t,w) is the original, undistorted image, n(t,w) is 0-mean, white Guassian noise process, and x*y represents the two-dimensional convolution of x and y.

The discrete representation of the image is obtained by sampling g(t,w) at a set of points on a Cartesian grid: $t=kT$; $k=-I/2, \ldots, I/2-1$; $w=lT$; $l=-J/2, \ldots, J/2-1$. Let $g_{i,j}$ ($0 \leq i \leq I-1$; $0 \leq j \leq J-1$), $f_{i,j}$ ($0 \leq i \leq I-1$; $0 \leq j \leq J-1$), $h_{k,l}$ ($0 \leq k \leq K-1$; $0 \leq l \leq L-1$), and $n_{i,j}$ ($0 \leq i \leq I-1$; $0 \leq j \leq J-1$) represent the discrete measurements of the observed image, the original image, the PSF, and the additive noise, respectively. The model equation for discretized images is

$$g_{i,j} = \sum_{k=0}^{K-1} \sum_{l=0}^{L-1} h_{k,l}\, f_{i-k,\, j-l} + n_{i,j}$$

If the dimensions of the spatial images, $\{g_{ij}\}$, $\{f_{ij}\}$, and $\{h_{k,l}\}$, are the same, we can transform the previous equation into a frequency domain representation. Let N and M be the row dimension and column dimension, respectively, of the three images. These dimensions can be selected arbitrarily, but the values selected should be greater than J+L-1 and I+K-1 in the horizontal and vertical directions, respectively, to prevent the convolution window from extending off the image in the previous equation. They can also be selected as a power of 2, so that efficient implementations with the Fast Fourier Transform (FFT) are possible.

If Fourier Transforms of the observed image $\{g_{ij}\}$, the restored image $\{f_{ij}\}$, and the PSF $\{h_{ij}\}$ are defined as G(u,v), F(u,v), and H(u,v), respectively, then the frequency domain representation of the constrained inverse filter is:

$$\hat{F}(u,v) = \frac{H(u,v)^*}{H(u,v)^*\, H(u,v) + \gamma}\, G(u,v)$$

where $H(u,v)^*$ is the complex conjugate of H(u,v), and γ is an arbitrary constant that controls the magnitude of the norm squared of the estimate of $\{f_{ij}\}$.

2.2 Tracking and Bandwidth Reduction--The tracking algorithm operates on a monochrome image to detect man-made objects in the field of view and track them over multiple images. The major elements of the tracking algorithm are multithreshold segmentation, boundary tracing, linearity filter, connected component analysis, and silhouette matching. Multithreshold segmentation is used to identify regions of the image with relatively constant intensity. A boundary tracing algorithm produces boundaries of regions that are passed through a linearity filter to determine which regions contain straight edges identifying them as part of a man-made object. Once the man-made pieces are assembled into objects by connected component analysis, tracking is performed by the silhouette matching algorithm, which compares silhouettes of objects in successive images to determine relative motion. Figure 1 is a data flow diagram of the tracking and bandwidth reduction algorithm.

The following subsections describe the algorithm for each component function of the tracking algorithm. Due to the large degree of commonality between tracking and bandwidth reduction, the latter is discussed here in subsection 2.2.8.

2.2.1 Window Average--The window average is a simple data-independent operation transforming the original intensity image into a smoothed intensity image. Its function is to help remove sensor noise in the image, ensuring that distinct regions in the image have relatively constant intensity. The operation computes a new value at each pixel position by averaging the input intensities of pixels in a window about the point. A 5 x 5 window was found to work well on test imagery.

2.2.2 Monochrome Segmentation--Monochrome segmentation divides the smoothed input image into different regions, where each region is characterized by a "nearly" uniform gray level. Labels are selected to represent different intensity ranges, and each region is appropriately labeled. This is accomplished in three major steps. First, an intensity histogram of the image is computed. Second, this histogram is searched for local minima and maxima, which define the intensity ranges corresponding to interesting regions in the image. Each distinct range is assigned a label and an upper and lower threshold defining the range. The third step is to apply the thresholds to the image, replacing the value at each pixel with the appropriate label. A list containing the labels and the average intensity values of their corresponding regions is also output for use by the bandwidth compression algorithm.

2.2.3 Boundary Tracing--Boundary tracing is a data transformation algorithm whereby encoded region boundaries are obtained from a labeled image. The encoded region boundaries (silhouettes) are much more compact than a complete image, requiring about four bits to store each image pixel that is on the boundary of a region. Conceptually, the algorithm interrogates each pixel of the image in an orderly fashion. At each point, the current pixel is examined to determine if it is on the boundary of a region that has not been traced yet. If so, the algorithm traces the boundary, ending at the same pixel where it began. The algorithm proceeds to examine the next point in search of more regions. When every image point has been examined, all of the regions have been traced.

2.2.4 Linearity Filter--A linearity filter is applied to the set of silhouettes. This filter computes a measure of linearity for each silhouette to determine if they correspond to man-made objects. The output is a binary image with nonzero values at boundary points of regions whose silhouettes were relatively linear.

The computation of the linearity filter is illustrated in Figure 2. Using a sliding window of width w, the angles s1 and s2 formed by the current point and the endpoints of the window are determined. The difference between these two angles is the curvature. For each silhouette, the linearity measure is the average curvature of all boundary points. If this measures exceeds a predetermined threshold, the boundary points of the region are set in the output binary image, indicating that the silhouette corresponds to a man-made object.

2.2.5 Connected Component Analysis--Connected component analysis is a general-purpose function that identifies connected sets of pixels in an image. Each connected set is identified with a label in an output labeled image and a location in an output feature file. In this instance, connected component analysis operates on the binary image output from the linearity filter to determine which segmentation silhouettes (and therefore regions enclosed by these silhouettes) belong to the same object. Since different components of an object may have different intensities, an object may be segmented into several adjacent regions. Since the regions are adjacent, they can be grouped into one component by connected component analysis.

2.2.6 Trace Components--This algorithm uses the starting locations and the labeled image output from connected component analysis to trace the boundary of each component (which in this case is a single target). Because the starting locations are known, this algorithm is considerably more efficient than the boundary tracing algorithm described in subsection 2.2.2. Thus, in this algorithm, it is not necessary that each image point be visited; however, the same method for tracing a region boundary is used.

2.2.7 Fast Silhouette Matching--Fast silhouette matching compares the silhouettes of targets found in the current frame to stored silhouettes of targets tracked in previous frames. Depending on which new targets match which previous targets, the tracked target information is updated, and any new targets are added to the track list. To match new and previous targets, each new silhouette must be compared to each old silhouette. The match scores are used to determine which new and previous targets correspond.

To compare one new silhouette with one previous silhouette, the (x,y) translation is determined, which maximizes the number of coincident boundary points of the two silhouettes. In theory, this is done by considering each possible (x,y) translation and counting the number of coincident boundary points. In practice, each point of the previous silhouette is considered as the point that ideally matches the first point of the target silhouette. Then the two silhouettes are traversed simultaneously, incrementing counters in a two-dimensional histogram. Each counter corresponds to the (x,y) translation necessary to make the new and previous boundary points in question correspond. On completion, the histogram is searched for the maximum. The maximum value is compared to the length of the silhouettes to determine the accuracy of the match. Essentially this is a variation of the Hough Transform. Choosing corresponding starting points is equivalent to making a hypothesis about the relative motion between images. Incrementing a counter is equivalent to "voting" for a particular (x,y) motion vector. The peaks correspond to the most likely motion vectors. The actual motion vector generally garners the most votes. The advantage of such a technique is its robustness and relative insensitivity to noise.

2.2.8 Bandwidth Reduction--This is a very simple operation that combines several intermediate tracking results to be transmitted to a remote location for storage or viewing. The essential information from the image is the target signature, complete with as much detail of the target itself as possible. The region silhouettes produced by the boundary tracing algorithm are processed by the linearity filter and marked indicating whether that silhouette passed the linearity filter or not. Then the bandwidth reduction operation can look at each region silhouette and select the ones that were considered part of a man-made object.

3.0 Knowledge-Based Image Interpretation Concepts

Knowledge-based techniques for image interpretation are more robust than conventional techniques because they can identify symbolic image features on the basis of incomplete or imprecise information obtained from the image. Classical techniques, in general, detect objects or specific features of objects from images on the basis of the degree of match between the actual features and a fixed, a priori model of the features. When the degree of mismatch is sufficiently great, as determined by composing the magnitude of a degree of match metric to a threshold, the algorithms reject the conclusion that the feature was observed. However, knowledge-based techniques do not categorically reject the hypothesis; they associate a "confidence factor" with the hypothesis transfer the positive and negative evidence obtained to date for that feature to a database. If evidence is found that either confirms or refutes the existence of the specific feature, the database can be revised. It is this characteristic of knowledge-based systems that makes them invaluable for image interpretation in unstructured environments, such as the Space Station construction environment. These systems have the capability of deriving conclusions from imprecise or conflicting sources of information and maintaining the history of deductive steps applied to reach those conclusions to permit optimal utilization of all available information at any single instant of time.

There are four generic categories of knowledge-based scene interpretation algorithms, which are:

1. Scene Labeling
2. Temporal Resolution
3. Context-based Resolution
4. Knowledge-based Feedback Control for Resegmentation

Each of these techniques will be explained in succeeding paragraphs.

3.1 Scene Labeling--The reliability and accuracy of each of the image processing functions tabulated in Table 1 will be enhanced with an understanding of the scene context for the image being evaluated. The context is deduced from a set of labels applied to the scene by a scene labeling algorithm.

Honeywell has developed a Reasoning Region Classifier (RRC)[12] to identify and test the knowledge pertinent to each of a specific class of regions. RRC is a production rule system, with explanation facilities, whose goal is to characterize image sub-regions of interest, based on vision system observable featues, such as region uniformity, texture smoothness, topological features, etc. This system is currently implemented for the classification of man-made and natural objects in air-to-ground imagery, but it could be easily modified to discriminate objects for the space scenario. The model of the scene structure is a hierarchical database, which has the label "entire scene" at the root, and is subdivided at each level of the hierarchy as the classification of scene objects becomes more specific. The search for the true classification of a specific region or object is performed on the subtree which has the highest confidence level based on the production rules.

3.2 Temporal Resolution--Temporal Resolution is a technique for the resolution of conflicts that result from region classification for region labels. It consists of the following steps:

1. Identify a sequence of frames which have been segmented into regions each of which desplay a region in the neighborhood of the candidate region R.

2. Determine whether the classifier result on the candidate region, say R, in the present frame, is consistent with the classifier results on the portion of the image corresponding to this region in past frames of the sequence.

3. Otherwise, modify the classifier result R by multiframe decision smoothing.

3.3 Context-Based Resolution--Context-based resolution conflict removal combines region information and relational context information from the current scene, for modifying classifier decisions that are inconsistent with the world model as represented in the hierarchical database.

Conflict removal is performed by detecting inconsistent configurations in the scene. The production rules that are used by the context-based resolution technique are based on a priori world knowledge.

3.4 Knowledge-Based Feedback Control for Resegmentation--A scene is composed of multiple regions that have different sizes and shapes. A single segementation algorithm may not suffice to properly segment all these regions. Appropriate choices of segmentors based on the ancillary information are crucial. Further, each algorithm has an associated set of parameters. Proper setting of the values of these parameters has a major impact on the algorithm regardless of how robust it is. For example, a large window size in noise smoothing techniques can blur the edges between two regions, thus, resulting in an erroneous region classifications. In this case, adaptive thresholding can remedy the problem.

There are two types of knowledge-based control methodologies that can perform equally well under variations of scene characteristics encountered in Space Station scenario.

3.4.1 Open-Loop Control--The first type of control is called open-loop knowledge-based control. This is a process that directs the low-level processing through proper image operator selection and the associated parameter value selection. Open-loop control processing governs simple data reduction tasks such as noise removal and region of interest selection.

The Open-Loop Control scheme derives the process goal from the information stored in Short-Term Memory (data obtained from the current image) and the knowledge base. The process goal is usually based on temporal and ancillary information such as previous frame processing results, the lighting conditions and a priori information for the radiometric and topological characteristics of the current scene. Rules in the knowledge base are used to derive the process goal. An example of process goal derivation is:

IF (mission goal IS (satellite detection) IN (high clutter area))

THEN ((locate region with two parallel linear borders) AND (remove high frequency noise))

Based on the derived processing goal, the selection control identifies the proper image operators with their associated values. The selection is also generated by the rules stored in the knowledge-base. An example of the selection rules is:

IF (remove high frequency noise)
THEN RUN (window average routine)

The process module then executes the selected image operators with the given parametric values. The output of the process is passed directly to the next low level process. The process module can also generate some knowledge, such as the image contrast, which can be used by other processing modules. This generated knowledge is fed into the knowledge-base for subsequent use.

3.4.2 Feedback Control--The second type of control is called the feedback knowledge-based control process. It is designed for governing complex low-level processes such as segmentation and color image enhancement processes.

Similar to the open-loop control process, feedback control derives the process goal from the knowledge in the knowledge-base and short-term memory, and selects the appropriate image operators and their parametric values. Then, the image operators are applied to the image and the results are passed to a process evaluation module. The process evaluation module determines the next processing step. The evaluation module either: 1) accepts the output and passes it to the next processing module, 2) feeds the image back to the same process module, recommending different image operators or parameter values for more refined processing, or 3) rejects the results and bypasses the process module.

The evaluation decision is also based on rules and information acquired from the scene stored in the knowledge-base. An example of a region evaluation rule is the following:

IF (Goal Size = small) AND (Goal Shape = rectangular)
AND (Region Size = large) AND (Region Shape = rectangular)
THEN (Resegment region with a lower threshold)

These knowledge-driven control processes make the best use of all available information about the scene. Each processing module can achieve the best possible performance in satisfying the processing goal. Therefore, a high performance scene analysis system can be developed by synergistically integrating the low level processing results.

4.0 Architecture Analysis for Parallelized, Multi-Processor Implementation of Knowledge-Based Algorithms

Previous sections of this paper have briefly touched on the key Space Station tasks which can benefit from knowledge-based vision processing. Details of specific Space Station vision functions and their implementation have also been discussed. In this section, we overview key architectural issues in developing a hardware architecture and software methodology for implementing these vision functions.

Developing real-time architectures for imaging systems is acknowledged as a difficult problem in many respects and remains a highly active research area. The key issues include: how to attain necessary and sufficient performance; how to program and maintain real-time systems; whether to use homogeneous or heterogeneous hardware; how to integrate processors with the environment; and how to develop planned/evolutionary approaches based on standards. A general solution to these central issues does not exist. Instead they must be revisited for each new application considered.

Statements of hardware performance requirements and capabilities usually are given simply in terms of the millions of operations per second (MOPS) needed for a set of functions or available from a system. A more critical measure of system performance would look at: operations per second (OPS) as a function of algorithmic requirements; power requirements; physical size and weight; and cost. In short:

Performance Measure = OPS(algorithm) / Watt cm^3 \$

Because transportation costs and limited space and weight budgets are key drivers in Space Station construction, the key elements of this metric should be throughput as a function of algorithm performed and total volume required to achieve this throughput. Weight and power are typically correlated to volume for a given technology, and the desire is always to minimize cost consistent with achieving functionality. Typically, computer architects design systems in an attempt to keep functional units (e.g., arithmetic logic units or multipliers) maximally busy because algorithmic performance requirements are specified in terms of the number of adds, multiplies, etc. Applying this approach to image processing architectures leads to designs in which 90%+ efficiency is achieved but on only 2-5% of the total processor hardware. Maximizing the throughput-to-volume ratio leads to more compact systems in which functional units are not necessarily fully utilized and is a logical approach for signal and image analysis architectures destined for the Space Station.

The tradeoff between using a heterogeneous or homogeneous processor architecture is a crucial tradeoff for any image processing system. The tradeoff is driven by algorithmic requirements as well as issues of system expandability, programmability, and flexibility. Current robust algorithmic paradigms for imaging systems subdivide the processing steps into various categories. An image understanding paradigm which has been useful in developing computer architectures is shown in Figure 3. This paradigm categorizes algorithmic functions according to data structures and processing functions. It is straight-forward, practical, and robust to directly translate such paradigms to hardware systems as indicated in Figure 4. Such an approach leads by nature to a heterogeneous architecture and from experience tends to minimize system volume. In general, more specialized hardware modules lead to a more compact system, but maximizing the throughput-to-volume ratio in this fashion must be balanced with expandability, programmability, and flexibility requirements in the Space Station application.

Two aspects of programmability become issues for real-time image processors. First, image processing hardware must be designed to utilize a high degree of parallelism at all levels to achieve high performance. The software methodology and tool set must provide adequate means to deal with parallelism and must bridge the gap between coarse grained high-level languages (e.g., Ada, FORTRAN, Pascal, etc.) and fine-grained machine languages (e.g., microcode). Any inefficiency in the translation or compilation process directly impacts the total system hardware requirement. Second, a software methodology intended for use with heterogeneous architectures must support all processor types in an integrated fashion. These are especially important issues in the Space Station setting where software development and maintanence costs will likely be the dominant portion of total imaging system cost.

The application environment affects imaging system architecture in many important ways. In the Space Station environment, factors such as fault tolerance and recovery, reliability, and testability are clearly important to safe and effective use of any mission critical computing equipment. In addition to these more or less generic considerations, very specific design details can be influenced by the environment. For example, electrical and radiation induced noise effects of the space environment lead naturally to consideration of optical interconnect for high data rate sensor channels. It is also logical to consider performing sensor specific preprocessing functions local to the sensor to reduce or eliminate channel induced noise. A broader environmental issue is the type and number of video sensors which can be active simultaneously. An architecture is needed which can readily switch between sensors and sensor types.

A final high-level issue with significance to the Space Station application is the ability of the selected architecture to adapt in an evolutionary fashion to evolving mission requirements. To achieve such an adaptation capability requires a so-called "open" architecture in which modules may be added or replaced. Designing an open but heterogeneous architecture is difficult in that each element of the architecture brings specialized interconnection, software, and other requirements. Maximum use of standards is a necessity to successfully developing such an open architecture.

A specific image processor implementation for Space Station applications has been developed and is reported in a companion paper [13]. This Video Image Processor (VIP) design is based on careful consideration of the broad issues discussed above and on the specific requirements of the image processing tasks and algorithms discussed in earlier sections of this paper. Over 150 architectural variations were analyzed using advanced computer modelling techniques. The result, illustrated in Figure 5, is a two-level architecture using special purpose high-performance image pixel processing hardware operating in a pipelined fashion combined with a distributed shared-memory multiprocessor. These two levels perform the image frame processing and combined region and symbolic processing functions from the taxonomy of Figure 3. The relatively low update rates specified for VIP allow the array processing and general purpose computing functions of Figure 4 to be combined in the multiprocessor.

Although the VIP architecture satisfies the essential processing requirements for the knowledge-based vision algorithms previously described and provides essential growth room, numerous architectural research areas with direct application to the Space Station remain to be explored. These include:

Sensor Preprocessing. Gallium arsenide technology provides the capability to integrate analog, digital, and optical interconnection circuitry monolithically. This capability may be used to advantage in Space Station sensor preprocessing by combining analog to digital conversion hardware, preprocessing logic (for noise suppression, detector compensation, and bandwidth compression), and high speed optical data channels on a single chip located at the sensor.

Programming Methodology. Two research areas relevant to programming heterogeneous signal and image processing systems are being explored by us. The first is a hardware array processor architecture designed to perform certain run-time resource management functions through special hardware constructs [14]. This approach can out-perform static (e.g., compile-time) resource allocation and leads to a more productive throughput-to-volume ratio then software-based dynamic allocation schemes. The second approach is a normal form language, IMP, which provides the programmer with manageable access to hardware parallelism rather then attempting to "hide" parallelism. This approach gives the programmer a homogeneous software environment for programming a heterogeneous system -- hardware modules may be readily added or modified within the context of IMP.

Cellular Architectures. Image processing architectures based on collections of simple cellular processors [15] hold significant potential for maximizing the throughput-to-volume ratio in space-borne applications. A new pixel-processing architecture based on a parallel recirculating pipeline (PREP) is under development by us. This architecture avoids the classical computation-I/O-memory balance problem to achieve high pixel-processing performance in an extensible and high-order language programmable fashion.

Evolutionary Architectures. One research effort recently completed by us involved definition of an integrated signal and image processing subsystem using hardware, software, and mechanical standards in an open architecture configuration. Our architecture research laboratory (ARL) combines a multiprocessor environment with special purpose hardware in just such a configuration and allows us to plan and rapidly execute new processor module and system development in an evolutionary fashion.

5.0 Experimental Results

For the Color Image Enhancement Algorithms, the performance of the three algorithms was evaluated by synthesizing degradations which might be encountered for real space imagery for the high-quality photographs and then quantitatively comparing the degraded images, after enhancement with each of the three algorithms, to the original image. A block diagram of the quantitative evaluation procedure is shown in Figure 6. For the Tracking and Bandwidth Reduction algorithms, a sequence of 8 frames at 1 second intervals were digitized from the Mission 41-C "Video Highlights" video tape, where the imagery depicts the SYNCOM satellite rotating in space near the Space Shuttle, shortly after deployment, against a cloud-covered earth background. The Tracking function's accuracy was empirically evaluated by comparing the algorithm's estimate for the two-dimensional change in location of the satellite (displacement vectors) to the best-guess at the actual displacement vectors, which were estimated by visual inspection of the gray levels of each successive pair of images. The maximum error in the estimated displacement vectors for the eight frame sequence was 1 pixel vertically and horizontaly.

Each of the Color Image Enhancement algorithms were evaluated for the task of restoring degraded imagery for a range of image degradations in order to accurately characterize the algorithm's performance in terms of an empirically derived model for its behavior. The Color Image Balanced Histogram Equalization algorithm and the Color Image Accentuation algorithms are efficient computational techniques for the restoration of dynamic range for color imagery. Here the dynamic range is the difference between the maximum intensity value and the minimum intensity value of the luminance image. A set of three degraded color images were generated, one which had a dynamic range equal to 50% of the original image's dynamic range, one with 62% dynamic range and one with 85% dynamic range. These images were then restored with the Color Image Balanced Histogram Equalization algorithm and the Color Image Accentuation algorithm, in that order. Mean-square-error masures were then employed to quantitatively evaluate the accuracy of the restorations. These mean-square-error measures were the output luminance image signal-to-noise ratio and the output chromatic signal-to-noise ratios. For all cases but one, the quantitative measures demonstrated that the Color Image Enhancement algorithms did restore full dynamic range to the test imagery and also did not distort the intensity or chromatic information of the images. Further details can be found in 2.

A few of the results of the experiments with the Color Image Histogram Equalization and the Color Image Accentuation algorithm are presented in Figures 7 through 10. Figure 7 is the original image. Figure 8 is the degraded image with a 50% reduction of dynamic range with respect to the original image. Figure 9 is the result obtained by processing the image with the Color Image Balanced Histogram Equalization algorithm and Figure 10 is the result of increasing the image's saturation after histogram equalization. Inspection of these images demonstrates that full dynamic range has been restored.

6.0 Conclusions

There are a multitude of applications where knowledge-based techniques may be employed to improve the performance of image interpretation algorithms, for space applications. Because the benefits of knowledge-based image interpretation algorithms; increased algorithm reliability and increased robustness, are of great importance in the unique space environment, it is apparent that any future architectural concept development efforts should take knowledge-based techniques into consideration.

7.0 References

[1] NASA/KSC "Space Station Mission Requirements Report, NASA/KSC Document Number MRWG-001, February 1984.

[2] Honeywell Systems and Research Center, Video Image Processor for the Space Station, Final Report, December 1986.

[3] Honeywell Systems and Research Center, Space Station Video Image Processor Concept Development, Final Report, 1985.

[4] D.M. McKeown, Jr., "Knowledge Based Aerial Photo Interpretation," *Photogrammetria* 39 pp. 91-123 (1984).

[5] M. Nagao and T. Matsuyama, *A Structural Analysis of Complex Aerial Phtotographs*, Plenum Press (1980).

[6] W.A. Perkins, T.J. Laffey, and T.A. Nguyen, "Rule-based interpreting of aerial photographs using the Lockheed Expert System," *Optical Engineering* 25(3) pp. 356-362 (March 1986).

[7] L. Sauer and J. Taskett, *Cuitural Feature and Syntax Analysis for Automatic Acquisition*, SPIE Conference on Processing of Images and Data from Optical Sensors (1981).

[8] M.D. Levine and A.M. Nazif, "Low Level Image Segmentation: An Expert System," *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6(5) pp. 555-577 (September 1985).

[9] Y. Ohta, *Knowledge-based Interpretation of Outdoor Natural Scenes*,Pitman Publishing, inc. (1985).

[10] J.J. Nitao, A.M. Parodi, "An Intelligent Pilot for an Autonomous Vehicle System", Proceedings of the 2nd Conference on Artificial Intelligence Applications, Miami, FL, December 11-13, 1985, pp. 176-183.

[11] B. Bhanu and W. Burger, *DRIVE - Dynamics Reasoning from Integrated Visual Evidence*, Proceedings of the DARPA Image Understanding Workshop (Feb. 1987).

[12] Honeywell Systems and Research Center, Bandwidth Reduction/Intelligent Target Tracker (BRITT) Demonstrator, Final Report, March 1984.

[13] Yalamanchili, S., et al, The Architecture of a Video Image Processor for the Space Station, Proc. of the NASA Workshop on Space Telerobotics, Pasadena CA, January 1987. (These Proceedings)

[14] Wehner, W.W., An Image Understanding System Based on Macro Data Flow, Proc. of SPIE Conference on Intelligent Robots and Computer Vision, Cambridge MA, October 1986.

[15] Preston, K., et al, Basics of Cellular Logic with Some Applications in Medical Image Processing, Proc. of the IEEE, Vol. 67, No. 5, May 1979.

Table 1. Cross reference between applications and algorithms.

	Color Image Enhancement	Tracking	Surveillance	Identification	Proximity Operations	Bandwidth Reduction
Construction	X	X		X	X	
Satellite Servicing	X	X		X	X	
Redezvous and Proximity Operations	X	X	X	X	X	
Inspection	X	X			X	X
Payload Delivery/Retrieval	X	X	X	X	X	X
Experiment Monitoring	X	X		X		X
Data Management and Communications	X	X		X		X
Training	X	X		X		X

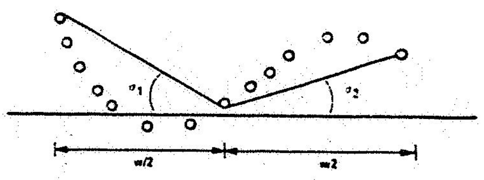


Figure 2. Computation of the Linearity Filter.

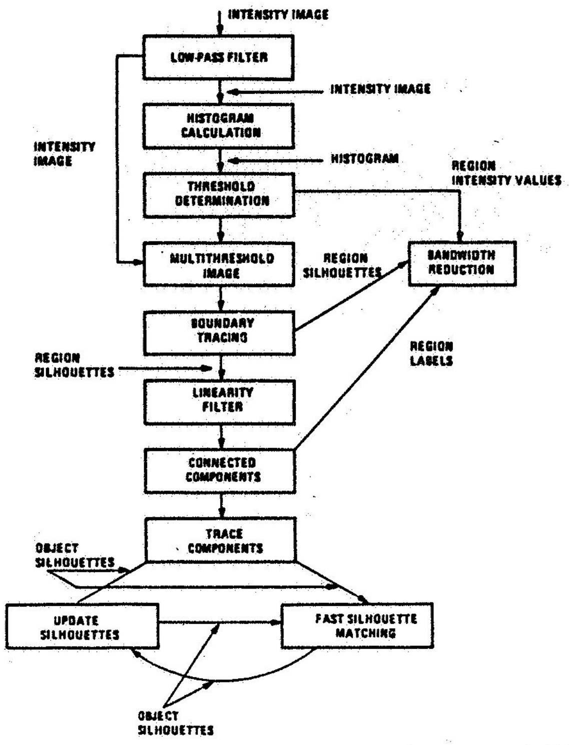


Figure 1. Functional decomposition of the Tracking and Bandwidth Reduction algorithms.

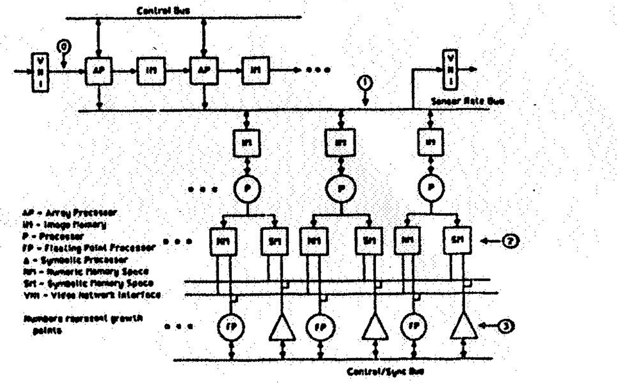

Control Bus
VNI
AP
IM
AP
IM
Sensor Data Bus
IM
P
NM
SM
FP
Control/Sync Bus
AP - Array Processor
IM - Image Memory
P - Processor
FP - Floating Point Processor
Δ - Symbolic Processor
NM - Numeric Memory Space
SM - Symbolic Memory Space
VNI - Video Network Interface
Numbers represent growth points

Figure 5. Video Image Processor (VIP) Conceptual Architecture.

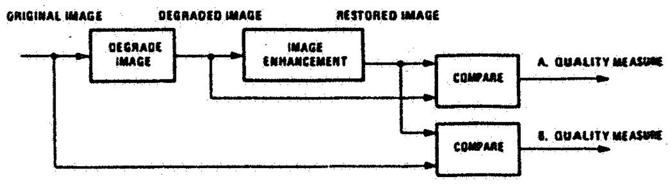


Figure 6. Color image enhancement algorithm performance is evaluated with various error measures.

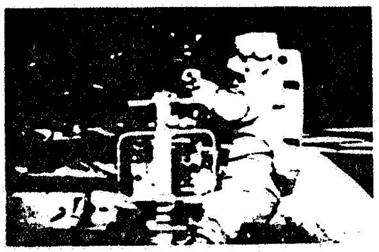


Figure 7. Original image.

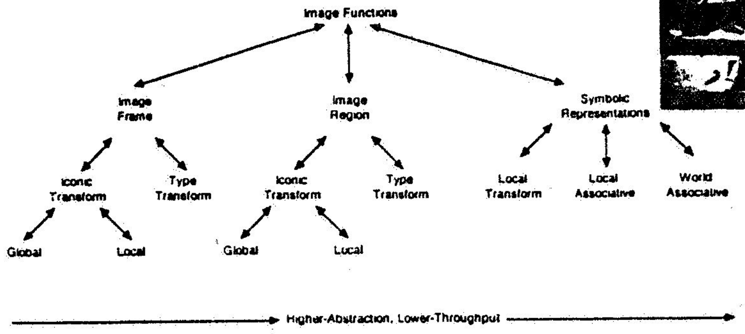
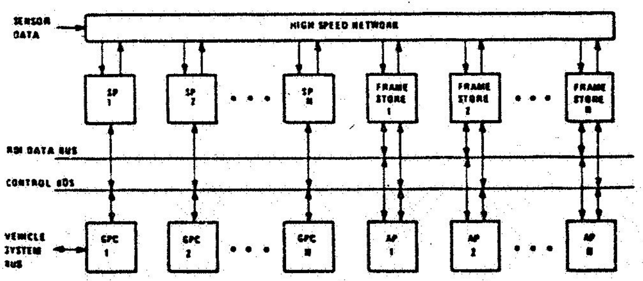


Figure 3. A Taxonomy of Image Understanding Operations.

Figure 8. Degraded image: 50% dynamic range.



- SP = signal processor
- AP = array processor
- GPC = general purpose computer

Figure 4. A Hardware Architecture Based on the Taxonomy of Figure 1.

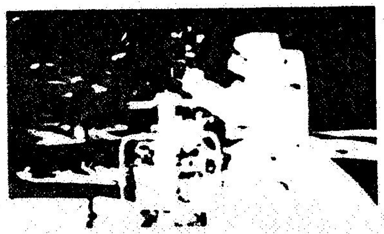


Figure 9. Restored image.

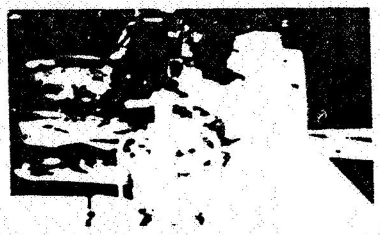


Figure 10. Restored and enhanced image.

53-04
3P
188123

Sensor Systems Testbed for Telerobotic Navigation

A.W. Thiele, D.E. Gjellum, and R.H. Rattner
Rockwell International Science Center
Thousand Oaks, CA 91360

D. Manouchehri
Rockwell International
Downey, CA 90241

RY 237991
RY 231002

1. Abstract

A testbed has been developed for the study of sensor systems to be used in telerobotic operations. The program, conducted in conjunction with Johnson Space Center of NASA, addresses the navigational problems associated with target acquisition and rendezvous for teleoperated robotic work stations. The program will utilize a mobile platform which will support various sensor systems during their development and testing in an earth-based environment.

~~2. Introduction~~

The testbed has been developed in support of a program to develop sensor systems that will aid in rendezvous and docking operations to be conducted as a part of the space station program. A mobile platform has been used to permit testing of these components in a conventional laboratory environment with consequent savings in cost and complexity. The sensor systems, while representative of devices currently in use for robotic applications, are not considered prototypical of the ones that will be used in the final applications. The test program provided information that will support the design of system augmentations and will lead to a comprehensive test program for sensor development.

3. System Description

The platform selected for this program, as shown in figure 1, is an electrically driven system utilizing three wheels which are steerable in a coordinated manner. It is capable of rapid changes in direction as well as turning in place. The internal control computer is capable of accepting commands to respond in real time to joy stick motion or to traverse a preselected path under program control. The commands are generated utilizing a host computer and transmitted over a radio frequency (RF) modem link.

The sensor systems address the problems of target acquisition, path planning and obstacle avoidance, and guidance. The systems are used in a teleoperated mode for the initial phases of the program with autonomous tests being planned for later in the program. The testbed has been equipped with an initial configuration of sensors and other components will be introduced at a later date.

The primary target acquisition system utilizes a video camera to permit target recognition and to provide azimuth and elevation information. The camera, as shown in figure 2, has automatic focusing and a variable focal length lens, so that target search can be performed in the wide angle mode and target tracking can be performed in the telephoto mode. A laser ranging device, shown in figure 3, is used to provide range information. The video camera and range units are mounted on a pan unit to facilitate the search function.

Obstacle avoidance is provided by a mapping sensor and an impact detection sensor. The mapping sensor, shown in figure 4, utilizes a commercially available ultrasonic transducer which is scanned over the full range of azimuth to simulate a radar mapping of the environment. The ultrasonic transducer is mounted so that it is facing down to a metal reflector, set at 45 degrees, which redirects the acoustic wave in a horizontal plane. The metal reflector is rotated by a stepping motor to provide the scanning function. A conventional radar system was not used for reasons of operational convenience, but the similarity in wavelength between ultrasound and microwaves should result in comparable resolution between the two transducer concepts. An impact detection system in the form of a skirt that surrounds the unit may be seen in figure 1. The sensors are small PVDF film elements attached to the circumferential band. These elements flex when the skirt impacts an obstacle and generate a small piezoelectric voltage which is amplified and detected.

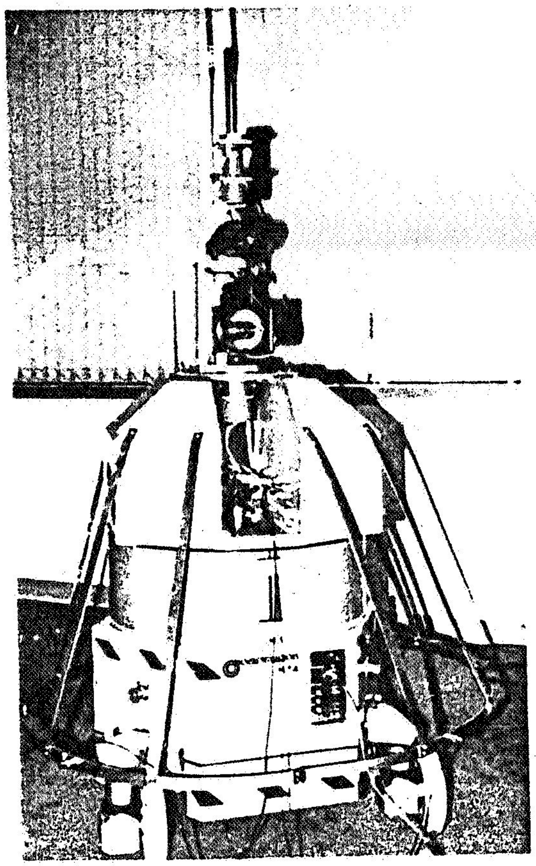


Figure 1
Mobile platform with sensors

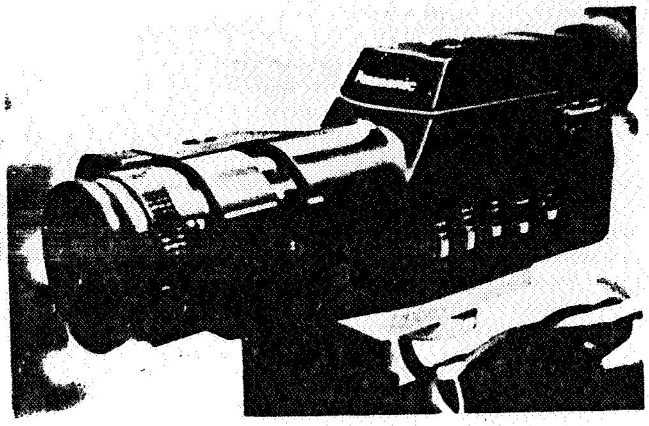


Figure 2
Video camera

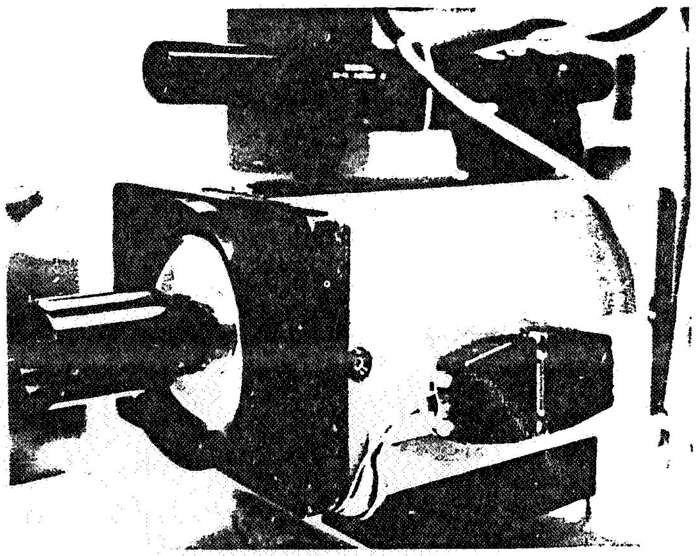


Figure 3
Laser rangefinder

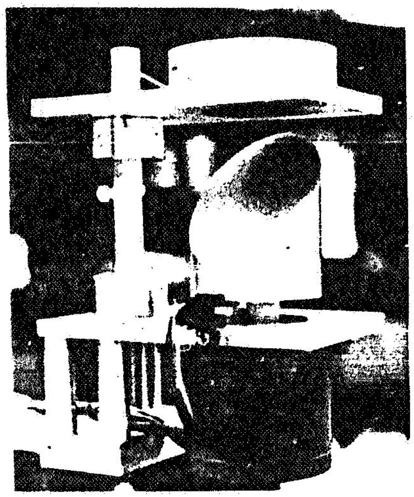


Figure 4
Acoustic mapping sensor

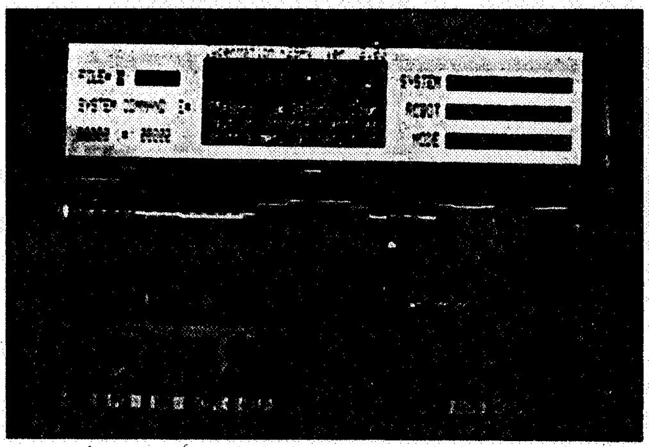


Figure 5
Monitor display

The data from the range, mapping, and impact sensors are interfaced to a microprocessor which controls, formats, and transmits the data to the host computer. The data is transmitted over the same RF modem as is used for the control commands of the platform. The presentation of the data on the monitor of the host computer is done using the available software in its resident system. A typical display is shown in figure 5. The acoustic range data is shown in graphical form with azimuth angle and range plotted on the same display. Parameters such as laser range, camera and laser azimuth angle, and acoustic range in the forward direction are displayed as digital values. The control of the platform is provided by a joystick at the workstation or through navigational instructions from the host computer.

4. Test Program

A test program has been carried out at the Johnson Space Center to evaluate the effectiveness of the sensors provided for navigation. The tests included operation in sight of the operator, operation using only the sensors, navigation around an obstacle and down a corridor, and rendezvous with a target. It was found that operation in an environment free of obstacles was easily accomplished using vision and range as the primary sensors. When obstacles requiring intricate maneuvers for their avoidance were introduced, the problem became more difficult. Real-time mapping of the space around the platform with an easily viewed display is required for navigation in this environment.

The vision system was by far the most important sensor. Augmentation of this capability by the use of multiple cameras to give panoramic coverage to simulate the capability of human vision would be desired. The test environment did not simulate the lighting conditions of a space environment and consideration should be given to that aspect.

The range information was not re-:ily accessible to the operator as he viewed the video monitor. An improved presentation of this data would be quite valuable. In lieu of this information, a stereo vision type of display could be considered.

The panning mechanism for the video and laser rangefinder units did little to improve the utility of the system. The ability of the platform to pivot in place using its own drive system was a far more valuable mechanism, since you could drive off in the direction that you were viewing at any time.

The mapping system was limited by the same consideration as the range system, lack of adequate display. The resident software in the host did not provide for a conventional range-azimuth display. One of our recommendations was to separate the sensor system from the host control system so that software for the display of the data could be developed separately. This change would also address a problem that was experienced with the RF link, that of contention and interference between the control and data acquisition functions.

It is our intent to continue development of sensor systems for this platform. Rockwell International has procured an identical platform for use in its Telerobotic Integration and Engineering Research Laboratory so that components may be evaluated prior to their testing at NASA.

This work has been supported in part by NASA under contract NAS9-17365. The support of Johnson Space Center personnel has been an important contribution to this program.

S4-74
78:
188124

Monovision Techniques for Telerobots

P.W. Goode and K. Cornils
NASA Langley Research Center
Hampton, VA 23665-5225

ND210491

~~1. Abstract~~

The primary task of the vision sensor in a telerobotic system is to provide information about the position of the system's effector relative to objects of interest in its environment. The subtasks required to perform the primary task include image segmentation, object recognition, and object location and orientation in some coordinate system. The accomplishment of the vision task requires the appropriate processing tools and the system methodology to effectively apply the tools to the subtasks. ~~This paper describes the~~ functional structure of the telerobotic vision system used in the Langley Research Center's ~~(LaRC)~~ Intelligent Systems Research Laboratory ~~(ISRL)~~ ~~and discusses~~ two monovision techniques for accomplishing the vision subtasks. is discussed as well as

2. Introduction

The telerobotic vision research objective is to adapt, develop, and evaluate noncontact sensing techniques to recognize and determine the location of objects in 3-space. To meet the objective, five goals have been established: (1) the techniques should be minimally complex in both hardware and software; (2) be generally applicable to a wide range of tasks; (3) require minimal or no alteration or premarking of the target objects; (4) be capable of mimicking a human operator (i.e., be able to provide target location information in terms of approach velocity as well as position); and (5) function in human real time (4 Hz.). An assumption that is allowed in order to minimize scene complexity is that the target objects are man made and a priori knowledge about them is available to the vision system. This is a reasonable assumption considering the nature of current and near future space operations.

3. System Configuration

The vision system is a distributed process within the Telerobotic System Simulation (TRSS) [1]. The system is functionally configured as two concurrent processes: the vision executive and the vision processor (fig. 1). The executive includes the functions of command interpretation, vision subtask determination, data base and modelling activities, local control activity, data conversion, and transfer of vision system status information to higher telerobotic system levels. The executive functions are performed by two modules referred to as the interpreter and the control interface. The interpreter directs the determination of target information by the vision system and the control interface processes and transmits the result to the telerobot's controller.

The interpreter's functions of command interpretation, subtask determination and sequencing, and data base organization and manipulation are hierarchical in structure and, therefore, are natural candidates for implementation as trees [2]. A tree is a collection of elements called nodes along with relationships among the nodes (e.g., parenthood, childhood, sequence, direction, precedence) that place a hierarchical structure on the nodes. A node can represent any entity (e.g., parent, child, subtask, shape, command) that does not violate the syntax or relational structure of the tree in which it exists (i.e. it must not impede the execution of the function). Trees can be subdivided into subtrees: A subtree consisting of shape nodes would represent an object, and one made up of command nodes would represent an execution imperative.

The vision interpreter is implemented as an abstract data type that allows the creation, deletion, and manipulation of trees of arbitrary size and function. The trees exist only at runtime and only when required to execute the requested function, thus, minimizing use of memory. As an example, assume that an imperative is received by the interpreter to locate a detected, but unrecognized object. The appropriate task tree is generated along with the necessary subtask, command, and object recognition subtrees embedded correctly in the task tree. The tree structure itself ensures the correct execution sequence. When the object is recognized, the recognition subtree is replaced by the object's description subtree known a priori, the location subtask subtree is generated, and the tree driven execution is performed again.

The control interface converts raw position data derived by the vision processor to a form compatible with the telerobot's control protocol. The TRSS data structure that handles dynamic system input/output is so constructed as to allow all position information to be accessed in terms of a common generic structure, generally referred to as an NSAP homogeneous matrix [3]. The matrix:

$$\begin{vmatrix} Nx & Sx & Ax & Px \\ Ny & Sy & Ay & Py \\ Nz & Sz & Az & Pz \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

is composed of an approach vector A describing the direction normal to the target plane, a sliding vector S denoting a direction normal to the A vector within the target plane and describing the rotation of the target plane about the A vector, the N vector which is the cross product of the S and A vectors, and the position vector P denoting the x, y, and z translations separating the axis systems of the camera and the target. The NSAP matrix contains all the information necessary to denote the orientation and position of the target with respect to the camera frame, and facilitates the various frame transformations that must occur in the telerobotic control process [4]-[5]. The angular parameters required for control can be extracted directly from the matrix. The decoupled angles used for finely resolved rate situations can be determined with the help of direction cosines as shown below:

$$\begin{aligned} &\text{rot. abt. } z = \arctan(Ny/Sy) \\ &\text{rot. abt. } y = \arccos(Az/(1 - Ay^{**}2)^{**}0.5) \qquad (1) \\ &\text{rot. abt. } x = \arccos(Az/(1 - Ax^{**}2)^{**}0.5) \end{aligned}$$

where checks for singularities and proper quadrants are implied. For position control situations or general system requirements, an NSAP to Euler transform has been implemented.

The vision processor performs the vision subtask as required by the executive and determines and advises the executive of the current status of vision processing. The vision processor is functionally segmented into low level, middle level, and high level processing. Low level processes include thresholding, gray level histogram generation and manipulation, and edge detection. Hardware and software implementing low level processes have generally been acquired from outside sources. Middle level processes include gray level based recognition, simple shape recognition, and target location. High level processes involve complex object recognition. Development and implementation of high level and middle level vision processes are the subjects of internal research. Two middle level processes that have been developed are discussed in this paper.

4. Monovision Methods

Two techniques that have application to the vision subtasks of segmentation, shape decomposition, recognition, and 3-space location are briefly discussed. The techniques are designed to extract 3-space information from a single two dimensional intensity image using prior knowledge and the principles of the perspective transformation.

The first method is based on the elastic matching [6] approach to pattern recognition and has application to shape decomposition, object recognition, and object location. It is an adaption of the linear programming technique of goal programming to the nonlinear problem of elastic matching [7]. Conceptually, elastic matching can be explained by envisioning a transparent reference image overlaying a goal image. The reference image is then warped or distorted to conform to the goal image by locally matching corresponding regions in the two images. The reference image is a flexible template that is modelled as a system of equation pairs where each equation pair represents a linear combination of patterns that a point in the reference image can describe in moving to a point in the goal image (fig. 2). The amount of displacement that each pattern contributes to the distortion is determined by identifying the values of the parameters Ai and Bi associated with each of the distortion patterns. The parameter values are derived by minimizing the absolute differences between corresponding reference and goal image points without violating the pattern constraints. This type of problem is easily modelled mathematically using the linear programming technique of goal programming [8]. The computational procedure that most efficiently resolves the optimal values of the goal programming model's parameters is the Simplex Algorithm.

The technique has been used to recognize simple three-dimensional objects of minimum curvature (i.e., near planar) and determine their location in 3-space. A single prototype shape (e.g., a rectangle) can be used to identify any of a primitive set of simple shapes by distorting it to match the image of an unknown shape. A simple shape is here defined to be a convex geometric figure formed on the surface of a sphere of large radius and the primitive set consists of rectangles, triangles, and ellipses. The values of parameters A3 through A5 and B3 through B5 yield information that allows recognition of the set members regardless of orientation. Once an object is identified, either as a simple shape or a combination of simple shapes, an exact model of its normal view is distorted to match the now known image, and information regarding its location and orientation can be derived from the parameters A0 through A3 and B0 through B3. Equations (2) through (7) show the geometric significance of the parameters.

$$\begin{aligned} A0 &= X' - X \\ B0 &= Y' - Y \end{aligned} \qquad : \text{translation} \qquad (2)$$

$$\begin{aligned} A1 &= -(1 - \text{gain}) \\ B1 &= -(1 - \text{gain}) \end{aligned} \qquad : \text{gain} \qquad (3)$$

where gain = X'/X or Y'/Y

A2 = (X' - X)/Y : rotation in x-y plane (4)
B2 = (Y' - Y)/X

A3 = -(1 - gain)/Y : perspective and triangular shape information (5)
B3 = -(1 - gain)/X

A4 = (X' - X)/a**2 : semicircular shape information (6)

B4 = (Y' - Y)/b**2

where a**2 = X**2 - Y**2 and b**2 = Y**2 - X**2

A5 = -(1 - gain)/Y**2 : elliptical shape information (7)
B5 = -(1 - gain)/X**2

Equations (8) through (10), which are based on properties of the perspective transformation [9], show the parameters' relationship to the range, pitch, and yaw respectively of the target object relative to the camera's axis system.

range = (f*Wo*(2 - A1))/((1 - A1)*Ws) (8)

where f is the focal plane distance of the camera/lens system, Wo is the object width, and Ws is the camera's image sensor width,

tan ϕ = 2*f*A3/(1 - A1) (9)

where ϕ is the pitch angle, and

tan θ = 2*f*B3/(1 - B1) (10)

where θ is the yaw angle.

Using a slightly different template (fig. 3), the technique has also been used to recognize arc segments and to decompose a geometrically complex object into its constituent shapes. The template is modelled as a system of n general equations of the second degree each of which represents a point on the arc segment of interest. The relative values of the derived parameters A, B, C, D, E, and F indicate the conic type of which the arc segment is a part (fig. 3) and their numerical values can be used to obtain the axis orientation, the foci, the vertices, the axis intercepts, and the eccentricty of the conic.

One way of determining a demarcation between simple shapes in an object's image is to locate boundary reversals (fig. 4). This is indicated when there is a rotation of axis between two adjacent arc segments such that the axes lie in diagonally opposite quadrants. The vertices of arc segments at the boundary reversals are used as end points of lines that subdivide the object's image into convex shapes that can be approximated by the primitive set.

By linearizing the problem, the computational efficiency of performing elastic matching is increased so that it becomes feasible as a real time procedure. Previous methods (e.g., exhaustive enumeration and dynamic programming) have required running times that are exponentially related to the number (n) of point pairs involved in the match:

T(n) = r**n (11)

where r is the number of possible global match configurations. For an n variable problem, the worst case running time of the Simplex Algorithm is linearly related to n:

T(n) $\propto$ n (12)

When the flexible template is transposed to its dual [7]-[8], each pair of points to be matched requires a variable. Thus, the addition of point pairs has little impact on the running time of the elastic matcher [7]-[8]. When using the technique for object location, the position update frequency is 4 Hz., which is in the realm of human real time (1.333 to 4 Hz.). It must be noted that most of the time in the position determination/manipulator activation cycle of the current testbed is consumed by the image processing activity and not by the parameter identification and location calculations. A faster image processor would allow frequencies approaching video frame rates (30 Hz.).

The second method determines the location and orientation of a planar object from any four points on the object that describe a reasonably convex quadrangle. Given the inter-vertex distances of the quadrangle and the optical parameters of the camera, the rotational and translational displacements between the object and camera can be uniquely determined.

The distance and orientation of the quadrangle relative to the lens axis frame can be solved in a closed form. The object points are defined as perspective projections of the image points along rays originating at the lens center, that is

Ti = Ki*Ii (13)

where the quadrangle <I0, I1, I2, I3> denotes the projection of the target <T0, T1, T2, T3> on the image plane (fig. 5). The axis system is chosen such that the x and y components of the projected image (Ix, Iy) lie on the image plane and Iz equals the focal length of the camera. In their paper on passive ranging, Hung and Yeh [10] prove that there exists a unique vector K which relates the target quadrangle and its image quadrangle and that it can be described in terms of the projected image points and the inter-vertex distances. The distances between the pairs of vertices can be described by a unique pair of nonzero real numbers, alpha and beta, independent of the coordinate system chosen, such that

$$I3 = I0 + alpha*(I1 - I0) + beta*(I2 - I0) \tag{14}$$

where noncollinearity implies that

$$alpha + beta \neq 1 \tag{15}$$

Equations (13) and (14) can be rewritten as

$$k3*T3 = k0*T0 + alpha*(k1*T1 - k0*T0) + beta*(k2*T2 - k0*T0) \tag{16}$$

By substituting for the Ti and dividing by k3, equation (16) can be transformed to

$$I3 = (k0/k3)*(1-alpha-beta)*I0 + (k1/k3)*alpha*I1 + (k2/k3)*beta*I2 \tag{17}$$

where the I vector represents the (x, y, z) coordinates of the image points. Noting that k3 is common to all the right hand terms, it can be considered a scaling factor that reduces the target quadrangle from its original dimensions to its projected dimensions at the image plane where k3 equals 1. Thus, from similarity, Hung and Yeh describe k3 in terms of the relationship of the magnitudes of the real and projected diagonals:

$$k3 = \left\|T0 - T3\right\| / \left\|(k0/k3)*(1 - alpha - beta)*I0 - I3\right\| \tag{18}$$

This information is sufficient to solve for the three dimensional positions of the quadrangle vertices (Ti) in the camera axis frame. The quadrangle orientation, described by the equation of the normal to the plane occupied by the quadrangle in 3-space, is determined by substituting the coordinates of any three vertices into the general equation of the plane. Solving the system of simultaneous equations gives the following explicit expressions for the orientation vector in terms of the quadrangle vertices derived above:

$$\begin{aligned} Ax' &= (T1y*T2z-T1z*T2y+T0z*T2y-T0y*T2z+T0y*T1z-T0z*T1y)/(D(T)) \\ Ay' &= (T1Z*T2x+T1x*T2z+T0x*T2z-T0z*T2x+T0z*T1x-T0x*T1z)/(D(T)) \\ Az' &= (T1x*T2y-T1y*T2x+T0y*T2x-T0x*T2y+T0x*T1y-T0y*T1x)/(D(T)) \end{aligned} \tag{19}$$

where

$$D(T) = T0x*(T1y*T2z-T1z*T2y)+T0y*(T1z*T2x-T1x*T2z)+T0z*(T1x*T2y-T1y*T2x) \tag{20}$$

and Ax, Ay, and Az are determined from Ax', Ay', and Az' by normalizing by the magnitude of the vector (Ax', Ay', Az').

Once the positions of the quadrangle vertices and the direction of its normal are known, the vectors that comprise the NSAP matrix can be found. The approach vector A is the orientation vector derived above. The sliding vector S is related to the slope of the base of the quadrangle with respect to the camera frame. It is the x, y, and z components of the vector T1 - T0 normalized by its length. The position vector P is simply the components of the selected point of approach on the quadrangle <T0, T1, T2, T3>. The intersection of the diagonals is commonly chosen.

For each probable target, it is necessary to determine and specify the alpha and beta parameters, based upon the inter-vertex distances of the quadrangle for each target introduced. One approach to entering new models in the data base is to automate this task in a one shot initialization procedure by processing one frame of the target image from a camera position normal to and at a known distance from the target. These parameters are calculated and stored in the data base. The calculations are based on equation (13) (and its transformations) with the K vector known. The results are presented here without derivation.

$$\begin{aligned} alpha &= V2/V1 \\ beta &= V3/V1 \end{aligned} \tag{21}$$

where

$$\begin{aligned} V1 &= I0x*(I2y - I1y) + I1x*(I0y - I2y) + I2x*(I1y - I0y) \\ V2 &= -(I0x*(I3y - I2y) + I2x*(I0y - I3y) + I3x*(I2y - I0y) \\ V3 &= I0x*(I3y - I1y) + I1x*(I0y - I3y) + I3x*(I1y - I0y) \end{aligned} \tag{22}$$

The raw state information consisting of the three translational and the three angular displacements of the target from the camera generated by both the elastic matcher and quadrangle projection methods is converted to the NSAP matrix. This matrix is input to the interface control section of the vision executive for further processing.

5. Future Work

The vision system development in the ISRL centered on the processing of single, two dimensional, intensity based (i.e., video) images. The next research phase will involve the extension of the system to process single three dimensional range based images as well as further refinement of the two dimensional techniques. The successful development of a laser vision sensor based on the FM-CW radar technique will support the next phase [11].

6. References

[1] F.W. Harrison, Jr. and J.E. Pennington, "System Architecture for Telerobotic Servicing and Assembly Tasks," presented at the SPIE 1986 Cambridge Symposium on Optical and Optoelectronic Engineering, Cambridge, Massachusetts, October 26-31, 1986.

[2] A.V. Aho, J.E. Hopcroft, and J.D. Ullman, Data Structures and Algorithms, Addison-Wesley, Reading, Massachusetts, 1983.

[3] C.S.G. Lee, "Robot Arm Kinematics, Dynamics, and Control," IEEE Computer, Vol 15, No. 12, December 1982, pp. 62-80.

[4] R.P. Paul, Robot Manipulators: Mathematics, Programming, and Control, MIT Press, Cambridge, Massachusetts, 1981.

[5] L.K. Barker, "Kinematic Equations for Resolved-Rate Control of an Industrial Robot Arm," NASA TM-85685, November 1983.

[6] B. Widrow, "The Rubber Mask Technique," Pattern Recognition, Vol 5, 1973, pp. 175-211.

[7] P.W. Goode, "A Multifunction Recognition Operator for Telerobotic Vision," Proceedings of the AIAA Guidance, Navigation, and Control Conference, August 1986.

[8] F.S. Hillier and G.J. Lieberman, Introduction to Operations Research, 3rd edition, Holden-Day, San Francisco, 1980.

[9] R.M. Haralick, "Using Perspective Transformations in Scene Analysis," Computer Graphics and Image Processing, Vol 13, pp. 191-221, 1980.

[10] Y. Hung, P. Yeh, D. Harwood, "Passive Ranging to Known Planar Point Sets," Proceedings of the IEEE International Conference on Robotics and Automation, pp. 80-85, 1985.

[11] F.E. Goodwin, "Coherent Laser Radar 3-D Vision Sensor," Proceedings of Sensors '85, November 1985.

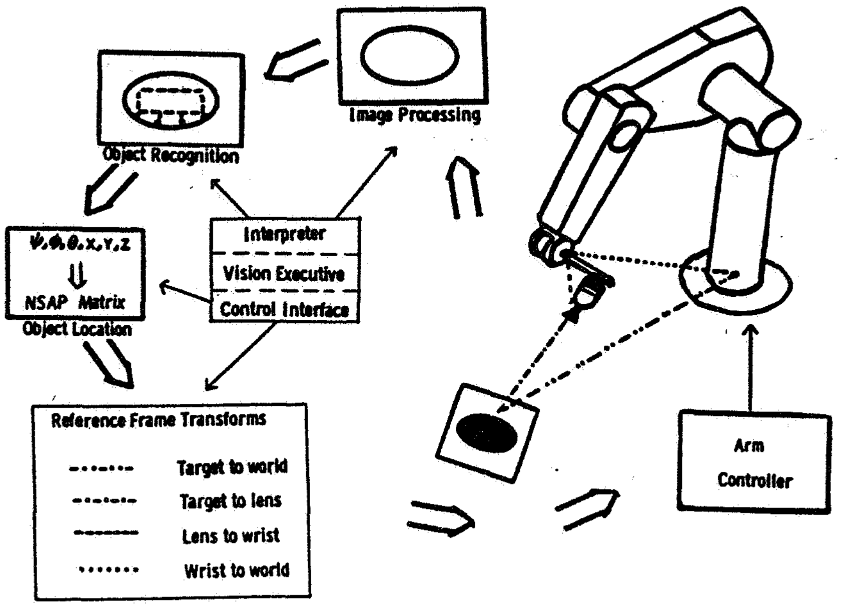


Figure 1. - System configuration.

A_0, B_0	: Translation
A_1, B_1	: Gain
A_2, B_2	: Rotation in X-Y plane
A_3, B_3	: Perspective of triangular shape information
A_4, B_4	: Semicircular shape information
A_5, B_5	: Elliptical shape information
$f(x, y)$	: Model function
$f(x^{*}, y^{*})$	: Image function

$$X + A_0 + A_1X + A_2Y + A_3XY + A_4(X^2 - Y^2) + A_5XY^2 = X'$$

$$Y + B_0 + B_1Y + B_2X + B_3XY - B_4(Y^2 - X^2) + B_5YX^2 = Y'$$

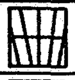
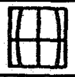
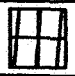
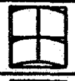
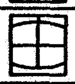
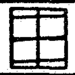
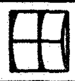
Distortion pattern	Distortion term
	XY
	XY^2
	Y
	Y^2-X^2
	YX^2
	X
	X^2-Y^2

Figure 2. - Elastic template.

MINIMIZE SUM OF ABSOLUTE DEVIATIONS OF MODEL POINTS FROM IMAGE POINTS

SUBJECT TO:

$$Ax_i^2 + Bx_iy_i + Cy_i^2 + Dx_i + Ey_i + F = 0 \qquad i = 1,n$$

NONTRIVIAL SOLUTION CONSTRAINT

$B^2 - 4AC = 0$	PARABOLIC ARC
DEGENERATE CASE	LINE
$B^2 - 4AC < 0$	ELLIPTIC ARC
DEGENERATE CASE	POINT
$B^2 - 4AC > 0$	HYPERBOLIC ARC
DEGENERATE CASE	CORNER
$\tan 2\theta = \frac{B}{A - C}$	THETA IS ANGLE OF CONIC'S AXIS WITH X AXIS

Figure 3. - Arc segment identification.

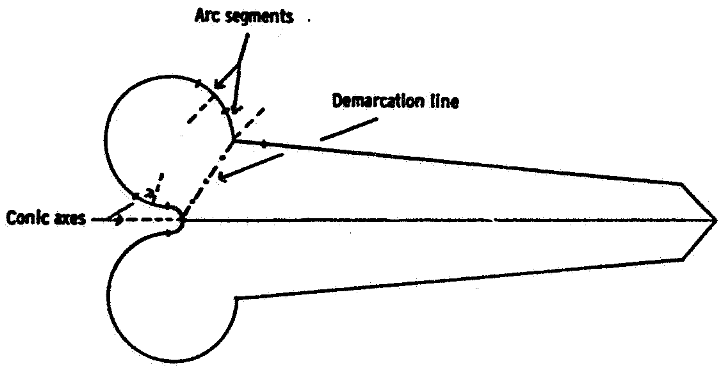


Figure 4. - Shape decomposition.

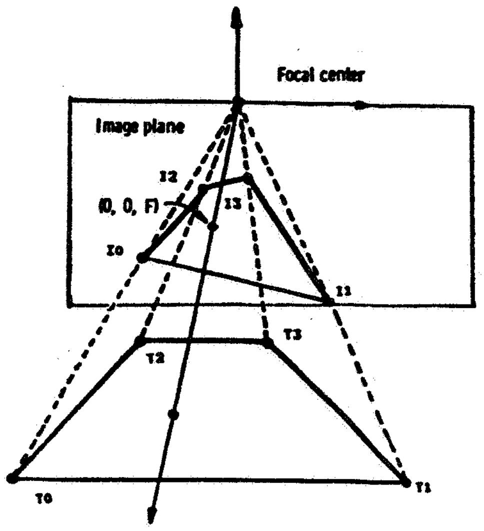


Figure 5. - Quadrangle projection.

55-63
138

188125

Tracking 3-D Body Motion for Docking and Robot Control

M. Donath, B. Sorensen, G.B. Yang, and R. Starr
University of Minnesota
Minneapolis, MN 55455

M 2 765962

1. ABSTRACT

A considerable number of activities require closed loop control of all six degrees of freedom of body motion. Accuracy and bandwidth are key issues in applications ranging from mirror positioning to robot control, from vehicular docking to autonomous construction and maintenance systems. One limitation has been the absence of a robust sensor capable of such measurement in a real time environment.

An advanced method of tracking three-dimensional motion of bodies has been developed. This system has the potential to dynamically characterize machine and other structural motion, even in the presence of structural flexibility, thus facilitating closed loop structural motion control. The system's operation is based on the concept that the intersection of three planes defines a point. Three rotating planes of laser light, fixed and moving photovoltaic diode targets, and a pipe-lined architecture of analog and digital electronics are used to locate multiple targets whose number is only limited by available computer memory. Data collection rates are a function of the laser scan rotation speed and are currently selectable up to 480 hz. The tested performance on a preliminary prototype designed for 0.1 in accuracy (for tracking human motion) at a 480 hz data rate includes a worst case resolution of 0.8 mm (0.03 inches), a repeatability of ±0.635 mm (±0.025 inches), and an absolute accuracy of ±2.0 mm (±0.08 inches) within an eight cubic meter volume with all results applicable at the 95% level of confidence along each coordinate region. plus or minus

The full six degrees of freedom of a body can be computed by attaching three or more target detectors to the body of interest. Structural motions can be tracked by attaching targets to the specific points of interest. The accuracy in reducing XYZ target position data to body angular orientation for this first prototype ranges from ±0.5 to ±1.0 degrees. Moving targets can be tracked at speeds exceeding 1 m/s with signal integrity tested but not limited to 25 Hz motions.

2. INTRODUCTION

Sensors that track body motion are critical to the implementation of closed loop position control systems in addition to facilitating the analysis of system dynamics. Many potential applications exist, from the analysis of human motion to the control of robot motion. Positioning accuracy and fast response of robot end-effector motion are important for many applications. While industrial robots have acceptable repeatability, their absolute positioning accuracy is relatively poor. This is for the most part due to their control architecture which involves closed loop position control at the joint level rather than at the end-effector level. There has been a need to accurately track the full 6 degrees of freedom of end-effector motion in order to facilitate what is known as end-point control (9). To enhance robot performance, a number of instrumentation systems have been developed for measuring 3D body motion in particular for the study of human movement. These include systems based on photogrammetry, electrogoniometry, sonic triangulation, accelerometry, and electro-optical phenomena. During the last decade, particular attention has been focused on electro-optical techniques. Within this group are several systems currently used in human motion laboratories: VICOM, SELSPLOT, CODA-3, and the United Technologies systems. Each system quotes comparable performance characteristics sufficient for the purpose of collecting human motion data, but each has inherent limitations in the maximum number of targets, in data processing time, in accuracy, or in bandwidth, which thus limit their utility to other applications. Other systems (2,7) have been developed for performance measurement of robot function but these are typically not compatible with closed loop control oriented sensing. Those that are (3) have other constraints.

The system described here, the Minnesota Scanner (or MnScan), has been under development for several years in an attempt to provide a cost effective, high performance alternative. Our present prototype, which was developed for tracking human motion with a goal of achieving 0.1 inch accuracy per axis for all three dimensions, has undergone extensive testing and calibration. Figure 1 shows the physical lay out of the laser scanning system in our Motion Analysis laboratory. This paper reports on our most recent results which demonstrate this system's potential for serving as a robot end point sensing system.

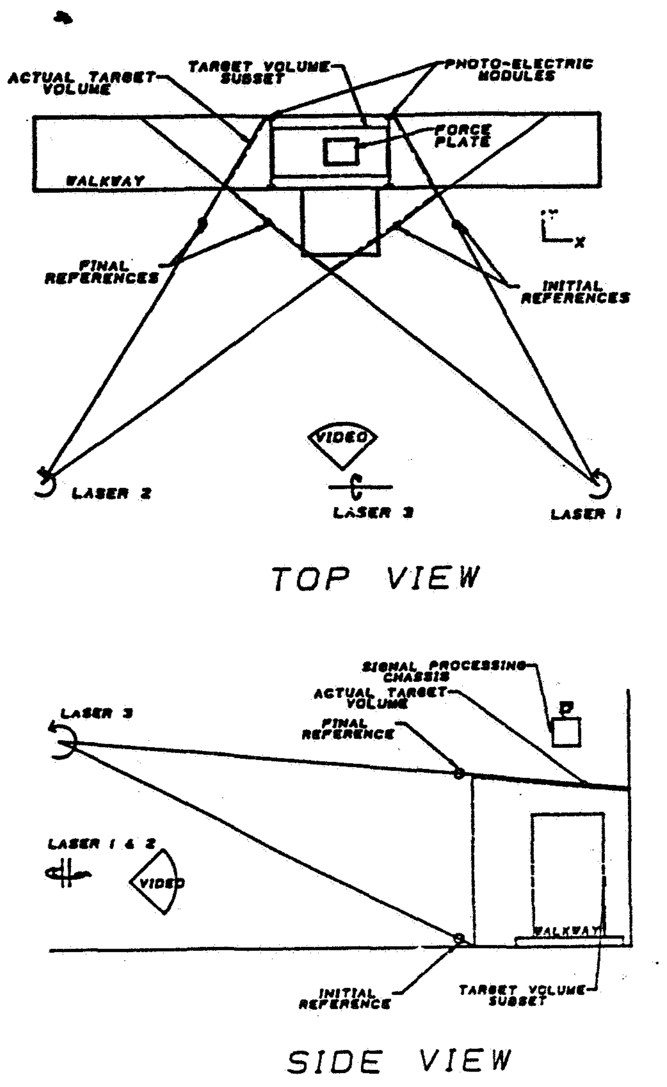


Figure 1: Physical Layout of the Laboratory

3. SYSTEM CONFIGURATION

The system is based on the concept that three planes of light, with differing normal vectors, intersect at a point in three-dimensional space. The basic premise of the method is that three intersecting planes define a point provided that none of the directional cosines of the three planes are the same. Thus, if the equations of three planes are known, the intersection point's coordinates can be found.

The planes are generated by passing light from a low power laser through a cylindrical lens arrangement and are swept through the measurement field at a constant velocity by reflecting the plane off of a mirror rotating at a constant angular velocity. The mirror is an octagonal prism attached to a 60 Hz reluctance synchronous motor. This allows for a high data collection frequency, 480 Hz, and precludes the need for phased lock loops and connections between the motors. As the plane moves through the work cell, it passes over photodetectors sensitive to the laser's wavelength. The signal from each detector is filtered to remove background light, amplified, and then further conditioned to provide a TTL pulse each time a plane passes over it. In our present configuration only one laser is in the volume at any one time. This is done by phasing the three mirror scan motors appropriately. This phasing, its effects, and other design issues, are discussed in more detail in MacFarlane, 1983 (5) and MacFarlane and Donath, 1985 (6).

In order to tell the lasers apart, three fixed initial references are located at one corner (lower right front) of the field so that each of these detectors views only one laser. The pulses generated by these three detectors form the basis for the data collection scheme. Every time an initial reference signal changes to positive, a counter is reset. These signals provide the data board with the current laser status so data can be routed correctly. In addition to the initial references, there are also three fixed final references which define the opposite end of the target volume and a number of target detectors, which are free to move within the target volume. The final reference detectors are not necessary but provide a means for compensating for any small deviations in the angular velocity of the mirrors, if necessary. Although closed loop speed control of the motors is possible, it was deemed unnecessary, given the present configuration and the desired specifications.

To locate a point in three dimensions, the following pieces of data are required; the equation of the plane when it passes the initial reference, which is assumed to be known, and the angle of rotation from the initial reference to the target for each of the three planes. The angle information is generated by sending the TTL conditioned signal from each detector to a data collection board. This board contains four major sections:

1. A 16 bit counter driven by an eight MHz quartz crystal.
2. A signal decoding section.
3. A bank of registers: 3-16 bit registers for each target detector and final reference detector.
4. A section to decode the register address for output.

and four register buses;

1. Memory input.
2. Enable to read.
3. Enable to write.
4. Memory output to parallel port.

Each time a detector "sees" a plane, the current counter value is placed in the appropriate register as determined by the signal decoding section. This is done continuously, transparent to the host processor every time a plane passes through the target volume. The value in the register represents the data board time for the plane to rotate from reference to target, and can be converted to an angle since the angular velocity is constant.

4. TARGET CONFIGURATION

Target photodetectors were selected to have high sensitivity in order to minimize needed laser power, and had a 120 degree wide angle field in order to be visible to all lasers as they moved in the field. The active area of the sensor was 20 mm^2.

As stated, each target detector defines a point in space. To define the orientation of a body, three points are required. A fourth point provides redundancy in the event that one detector is momentarily blocked from the view of the light planes, and also allows a least squares estimate to be fitted based on 4 sets of three points. The greater the separation distance between targets, the greater the accuracy in determining the body orientation. However, in order for the target mounts not to get unwieldy, the separation distance was constrained.

5. SYSTEM CONFIGURATION EFFECTS ON COMPUTATION

After the timing data has been obtained it must be converted to three-dimensional position information (points) in order to be useful. To locate each point the information required is the location of the axis of rotation of the plane, the location of the initial reference, and the angle of rotation from the initial reference to the target for each plane. With this information, each plane can be represented at the target, and the intersection of these planes define the target's (x,y,z) location. The exact equations can be found in Sorensen, 1986 (12).

In order to compute a target's location, the locations of the reference targets and the optical axes of rotation must be known. The axes of rotation are positioned so that they are parallel or orthogonal to each other and to the global coordinate system (GCS), with two axes being vertical and the third, horizontal in orientation. This is done to simplify the position calculations. The simplification is necessary in order to decrease the computational time required of data reduction. Bechtold (1) showed that if the axes of rotation were arbitrarily located, the solution process would be an open form iteration. To obtain a solution, all three angles are entered into the solution matrix along with an initial approximation for the solution. A Newton iteration method is then used to converge to the final solution. However, by orienting two of the light planes along a vertical orientation, their intersection yields a vertical line. This line defines the coordinates of the point in the horizontal plane. To define the third coordinate, the third plane is used to intersect the vertical line. The solution process in this case is closed form in nature and very short; only two equations. The derived solution in this case is thus reduced to two two-dimensional calculations per three-dimensional point. The time saved is immense and makes the process feasible for real-time situations.

6. DETERMINATION OF SYSTEM PARAMETERS

As indicated earlier, the location of each axis of rotation must be determined in order to effectively use the system. Other parameters that must be identified are the location of the initial and final reference photodetectors for each laser. To locate the positions of the six fixed reference diode targets and a local coordinate frame for each moving axis of rotation, a ZEISS precision measurement system was used. This measurement device consists of two theodolites interfaced to a computer. Once the theodolites are calibrated to provide their locations with respect to a global coordinate frame in the room, points can be located with respect to the defined frame with accuracy and repeatability better than 0.003 inches.

To do the system calibration, four or more points of known location are needed to calibrate the theodolites. These points were located on an optical bench (2m x 2m) which straddled the target volume during the calibration process (see Figure 2). This removable test bench was used to define the global coordinate system and to set up the ZEISS calibration points which were located with a six-foot vernier caliper. After the reference detectors

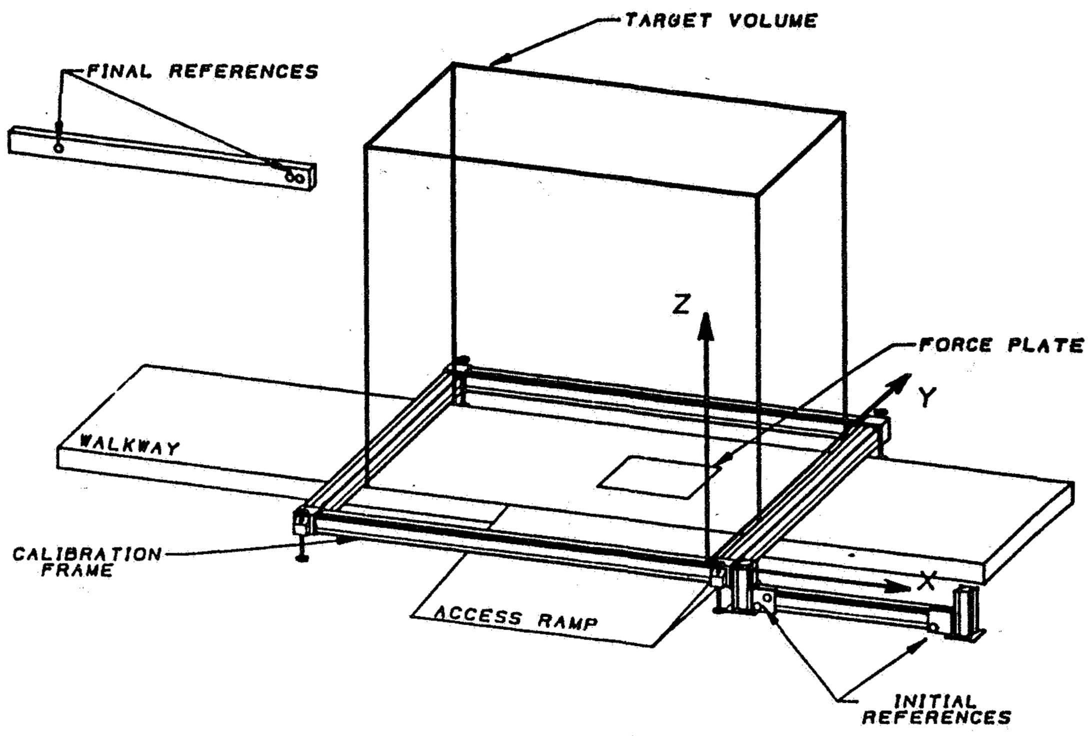


Figure 2: Target Volume as Defined by the Inertial References and the Global Coordinate System

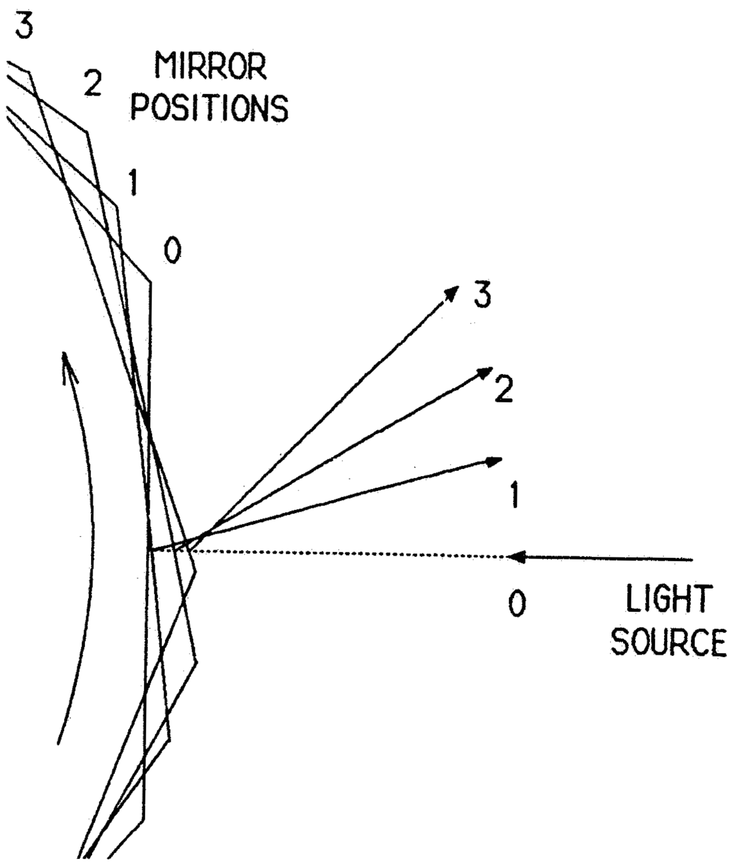


Figure 3a: Magnified View of Three Reflected Light Planes

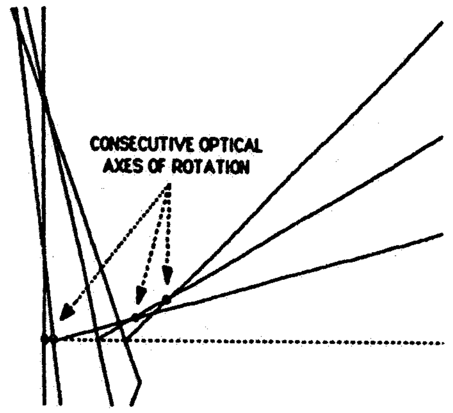


Figure 3b: Blowup of Region Adjacent to Moving Mirror

are affixed to the measurement field boundaries and the axes of rotation oriented, the three mechanical axes of rotation and the six reference detector locations may be measured. With all system parameters defined, the data generated by the system can be reduced to three-dimensional points.

7. MOVING AXIS OF ROTATION

Although the use of the octagonal mirror gives the system a high bandwidth, it introduces a complication. Since tha reflective surface is a finite distance from the mechanical axis of rotation, the optical and mechanical axes are not coaligned, and as a result, the optical axis is not fixed in space, but moves as the mirror rotates (see Figure 3). This motion is not negligible and cannot be ignored since the axis can move as much as 0.25 inches in each direction perpendicular to the axis of rotation. Fortunately, this motion is a function of realizable parameters and can be fitted to an equation. The determining factors are the geometry of the mirror and the line defined by the incident laser beam. A detailed model was derived from differential geometry and implemented in software (12).

8. STATIC PERFORMANCE TESTING

To determine if the data generated by the system is accurate, a detailed calibration was performed. The process involved moving a vertically-oriented, linear array of eight targets through the target volume.

The optical bench was used as a means for mounting and providing regular movement of the target configuration. A movable horizontal rail was placed parallel to the x axis to yield changes in y. Displacement of a vertical rail with the attached target configuration yielded changes in x. With the eight targets at varying heights, it was not necessary to change the z component. The targets were moved along six inch intervals in the x and y directions by the appropriate adjustment of the horizontal and vertical rails. This protocol yielded data on 616 discrete points within the target volume; eleven locations in the x direction, seven in the y, and eight in the z. At each location, data was recorded using both systems; the theodolite based method and the Laser Scanning approach. Each target was sampled (measured) 200 times by the Laser Scanning system to provide data for statistical analysis.

9. RESULTS

Accuracy: In testing the system a phase lag error was discovered. By modeling the error, based on a smaller subset (200 out of 616) of the total measurement, the phase lag was eliminated. The spatial locations of the detectors for the ZEISS and the MnScan system are shown in Figure 4 for 3 slices through the field. It can be seen that on the scale of these plots, the data from the two systems effectively superimpose. However, further analysis shows that there was a root mean squared error of 0.04 inches that remained and that a 95 percent confidence interval includes an error of ± 0.08 inches along each axis. These results were obtained at 480 Hz data acquisition rates. (Two data points are missing from Figure 4 because their associated data file was irretrievably lost by an inadvertent deletion. These were thus not included in any error or calibration analyses.)

10. RESOLUTION

The spatial resolution is dependent on the position within the target volume. The input parameters to the solution matrix, the three rotation angles, have an angular resolution; thus, the distance from the target to each laser affects the tangential resolution of each laser.

Each angle has a resolution determined by the angular velocity of the mirror and the clock speed on the data collection board. The angular resolution is the amount of rotation per clock count. For the present configuration, with an eight faceted mirror rotating at 60 Hz and an eight MHz clock, the resolution is:

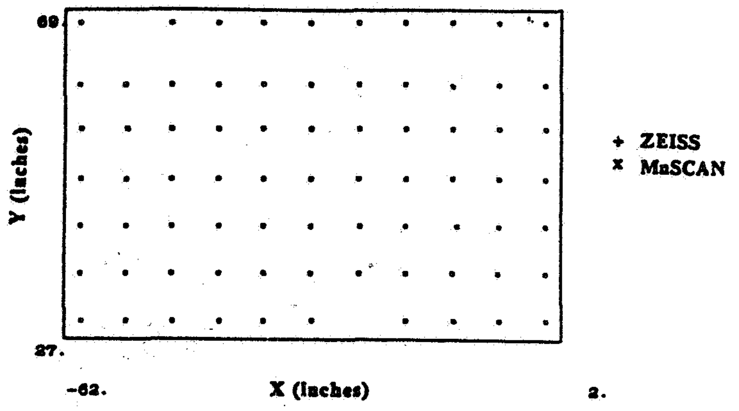


Figure 4a: Plot of Actual and Measured Points in the XY Plane After the Angular Correction

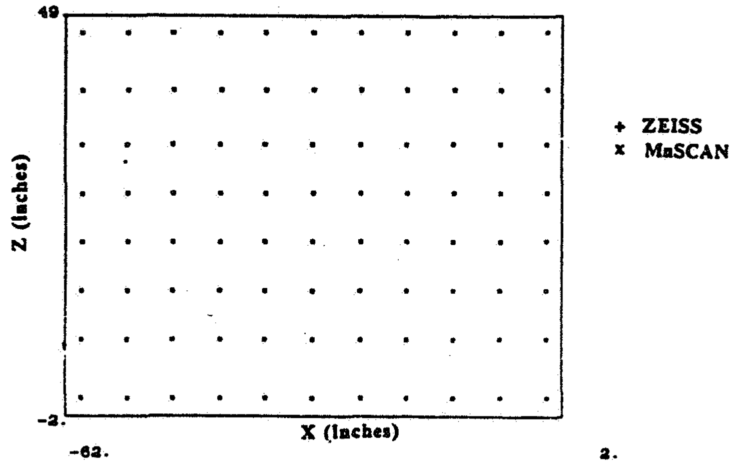


Figure 4b: Plot of Actual and Measured Points in the XZ Plane After the Angular Correction

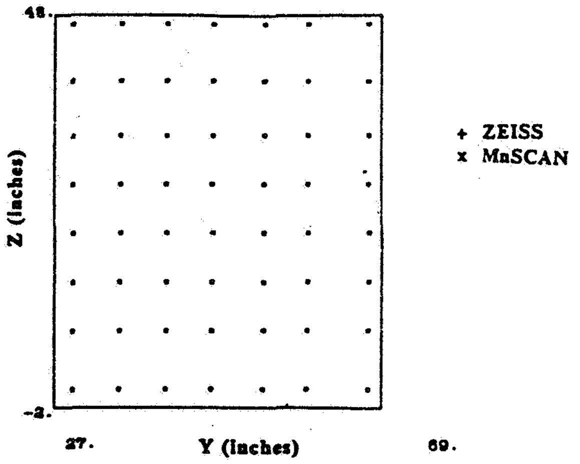


Figure 4c: Plot of the Actual and Measured Points in the YZ Plane After the Angular Correction

$$\frac{\left(90\ \frac{\text{degrees}}{\text{facet}}\right) \times \left(8\ \frac{\text{facets}}{\text{revolution}}\right) \times \left(60\ \frac{\text{revolutions}}{\text{second}}\right)}{8\text{E6}\ \frac{\text{counts}}{\text{second}}} = 0.0054\ \frac{\text{degrees}}{\text{count}}$$

At a distance of 30 feet (the maximum target distance for any laser) the angular resolution corresponds to a tangential resolution of 0.034 inches. This result is misleading because it is not the spatial resolution. By using the tangential resolution and the system's geometry, the actual spatial resolution at a point can be determined. This resolution has components along each of the three different axes which varies from half to two-thirds of the tangential resolution of 0.034 inches across the measurement field.

11. SIGNAL TO NOISE RATIO: POWER SPECTRAL DENSITY

Noise in the signal comes from a variety of sources including electronic, optical and environmental effects. If one were to examine the timing data from a stationary detector for one laser, the expected result, assuming no noise, would be a constant number. The actual result varies from five to ten counts depending on the angular location within the field; the closer to the initial reference the smaller the variation. To determine the impact of this noise, data was collected at 480 Hz to allow processing by a 10-bit FFT (Fast Fourier Transform) for signal decomposition. The resulting frequency magnitudes displayed a single spike at zero hertz and no signal elsewhere. To observe the low level noise the zero hertz value was reset to zero and the magnitudes were replotted (see Figure 5). Now, peaks appear at 60 and 120 Hz with the 120 Hz peak having a magnitude 0.03 percent of the zero hertz magnitude. This result is the same throughout the field for each plane of light, independent of detector and signal processing channel. We have identified the source of the 60 Hz noise and are working towards eliminating it.

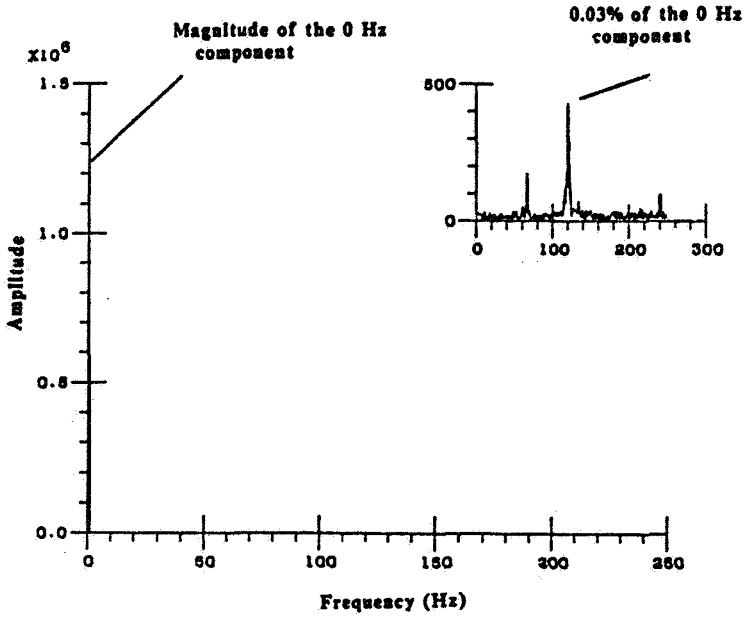


Figure 5: PSD of a Timing Signal

12. SCATTER PLOTS AND NOISE FILTERING

To examine the noise effects on accuracy and resolution, a set of data (300 samples at 480 Hz) for a stationary target was collected and reduced to (x,y,z) coordinates. This data is represented in two planar projections, an (xy) plane view and a (yz) plane view. The (xz) plane is not used because x and z are independent and the plot of this plane would just show up as a cloud of points. An important fact to remember is that the active area of the detector is 20 mm^2, or 0.176 inches in the x and z directions. The results indicate that all calculated points fall within the active area.

The planar projections (Figure 6) show the effect of the 5 to 10 counts of noise at a particular spatial location within our field. In both projections patterns are visible. For the (xy) plane, clusters located at the intersections of a diamond shaped grid are evident. This happens because the rotating light planes from lasers one and two pass the detector at discrete values associated with the clock resolution. The diagonal lines of the grid represent the orientation of the light plane at each discrete value. The small clusters in the (XY) scatter plot are due to a variety of effects that are beyond the scope of this paper. Suffice it to say that they are not a function of motor speed or motor phasing and are thus not easy to remove at their source. Nevertheless, despite this noise, the resulting error over the test region falls within + 0.08 inches at the 95% level of confidence, well within the design specs for this prototype.

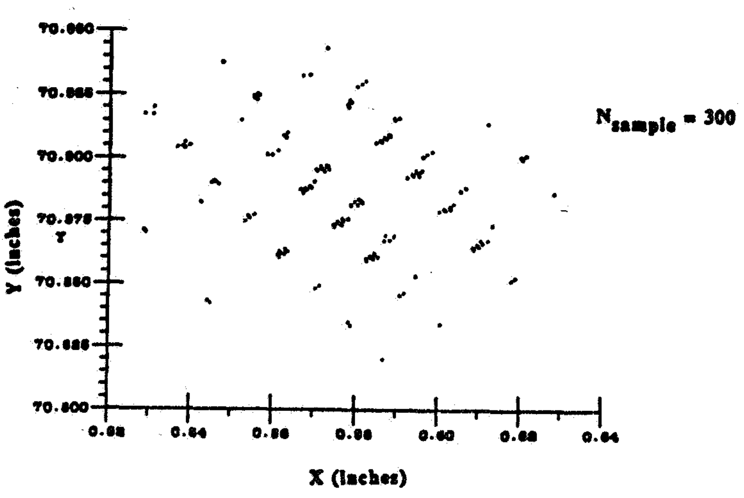


Figure 6a: Scatter Plots of the Signal Noise Effects in the (XY) Plane

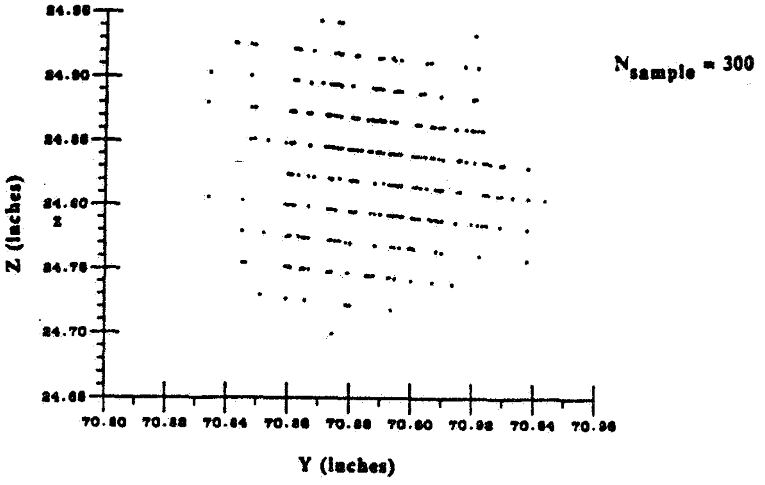


Figure 6b: Scatter Plots of the Signal Noise Effects in the (YZ) Plane

Although these projections show the locations of each digitized point, no information on relative density i obvious because multiple points are superimposed. This is important because in the (xy) projection of Figure 6 only 20 percent of the data points are visible. To determine the distribution of the computed position, a tw dimensional histogram of the (xy) scatter plot was made. The area was divided into equal regions such tha groupings were preserved, and the number of points in each cell was totaled. This data was then plotted in orde to produce the histogram in Figure 7. This graph shows that the points indeed distribute themselves alon quantized laser planes.

To examine low pass effects on the accuracy and repeatability, the timing data can be passed through digital filter. The filter used was a tenth order Butterworth lowpass with time reversal to preserve phase. T prevent start up oscillations, the first half of the signal is folded and inverted about the beginning of th signal and likewise for the second half of the signal. This allows smooth transition signals and any start u disturbances should die out in the artificial portion of the signal. To provide design variability, the -3d cut-off level can be specified to achieve the desired performance. To eliminate the 120 Hz peak, a 100 Hz -3d cut-off filter can be used. Each of 27 sets of modified timing data was passed through a filter and then rounde off to preserve the discrete nature of the data for this example. At this point the filtered timing data can b reduced to coordinates and plotted. These results (Figure 8) have the same patterns as before but now the nois has been attenuated, as expected. Although the noise has diminished, the signal to noise ratio can be improve even further if desired. The result of using a filter with a -3db cut-off level of 40 Hz is shown in Figure The histogram indicates that the filter reduces the group of points to within a range of 0.04 inches in the x an y directions or an error of ± 0.02. The same process yields a range of 0.06 inches in the z direction (error o ± 0.03).

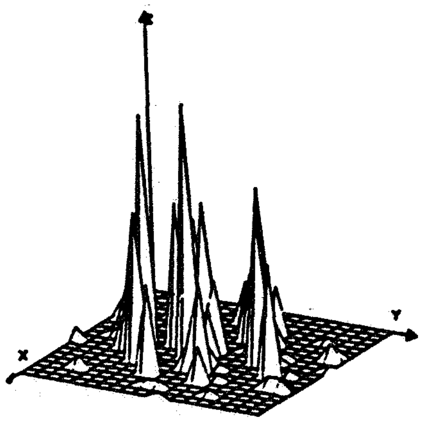


Figure 7: Histogram of (XY) Plane Scatter Plot

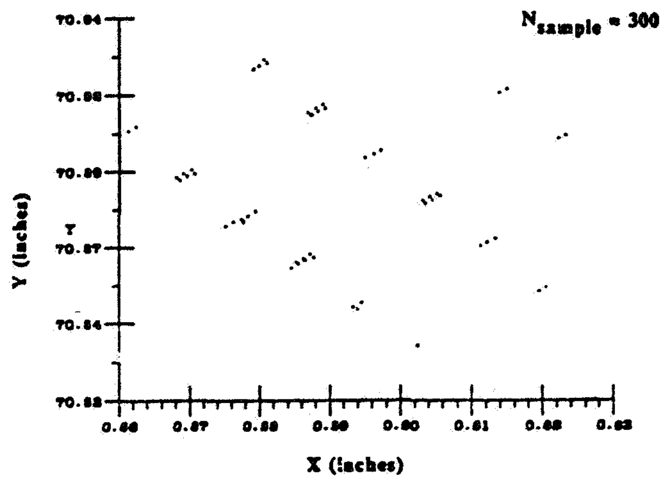


Figure 8a: Scatter Plots of the Signal Noise Effects in the XY Plane with a 100 Hz Low-Pass Filter

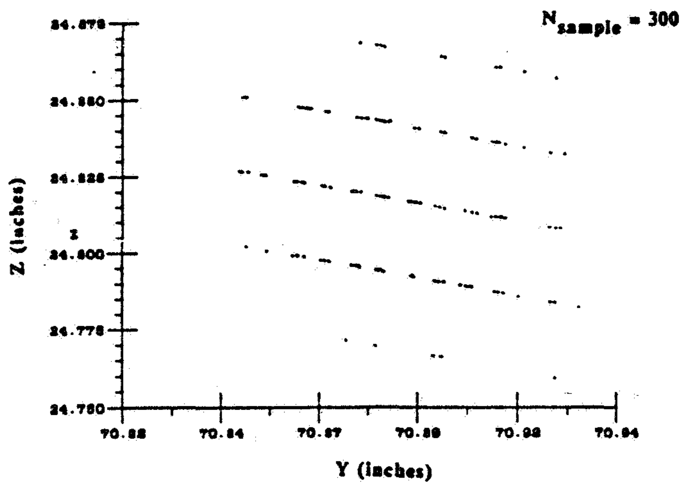


Figure 8b: Scatter Plots of the Signal Noise Effects in the YZ Plane with a 100 Hz Low-Pass Filter

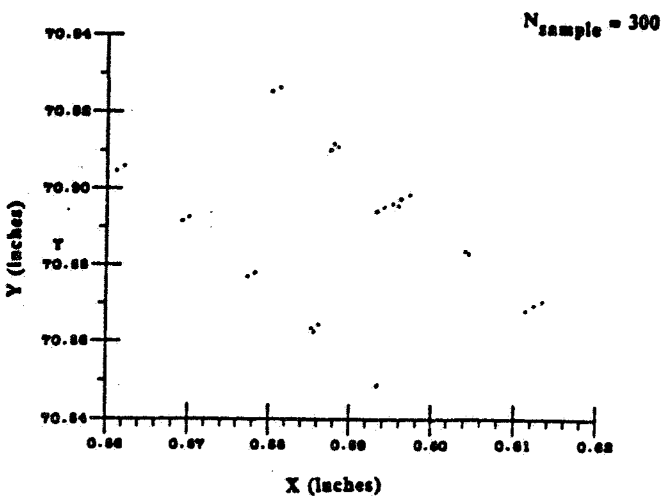


Figure 9a: Scatter Plots of the Signal Noise Effects in the XY Plane with a 40 Hz Low-Pass Filter

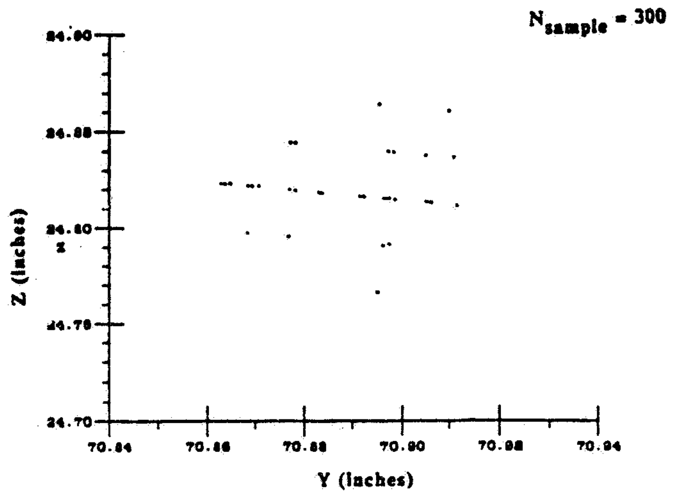


Figure 9b: Scatter Plots of the Signal Noise Effects in the YZ Plane with a 40 Hz Low-Pass Filter

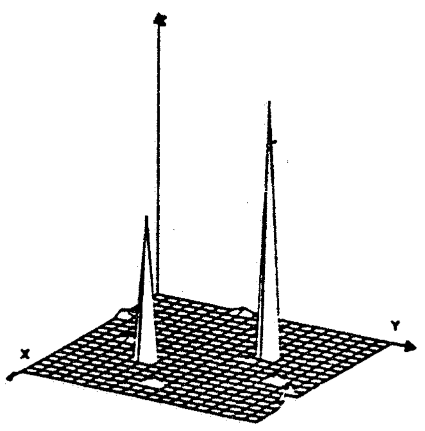


Figure 9c: Histogram of (XY) Plane with a 40 Hz Low-Pass Filter

13. DYNAMIC TESTING

Two additional tests were implemented to test the system's performance. The first test considered a single pendulum supporting the eight targets on a 1.75 meter bar. The resulting low frequency motion covered an entire planar section of the target volume in an oscillatory pattern. Data was collected at frequencies from 60 to 480 Hz and showed, as expected, that there were no local discontinuities or global warping within the target volume.

The eight targets were placed at relatively regular intervals along the bar with the farthest being about 72 inches from the pivot point. The targets were in a linear configuration to achieve the effect of concentric circles. The advantage of this configuration is that the angular position, with respect to the center (or the axis) of rotation, is the same for each detector. This fact was used in two separate analyses of the pendulum data.

The first test was a simple plot of the points reduced from the timing data (no filtering). Since the motion was in the (xz) plane, only two axes were used for the plots. In the two cases displayed (Figure 10), the sampling frequencies were 120 and 480 Hz. Other sampling rates were observed but, since the results were similar, only these two are shown. On these displacement records, the concentric arcs are clearly evident. At this point, inspection of these graphs tends to support that there are no irregularities in the target volume. Furthermore, the arcs traced out in the 480 Hz case are sharply defined and very smooth, which indicates that the signal noise is indeed small in the global sense. In the 120 Hz case, the arcs are still very smooth, but not as sharply defined. This is partly due to the longer sampling period and, thus, multiple swings of the pendulum. The small effects of signal noise and the dynamics of the pendulum and its mount are more than enough to cause this blurring. The dark spots are the places where the pendulum changes direction; the deceleration and then acceleration puts the pendulum in one place for a longer time, thus causing the sampling density to be higher in these locations. Since there is friction in the pin joint, there will be damping of the motion; this is reflected by the decreasing amplitude of each swing.

A second method of analyzing the pendulum data was to plot the computed angular position of each detector with respect to time. This method produced excellent results and is described in Sorensen, 1986 (12).

The second test was to determine performance under small displacement and high frequency conditions. A single target was attached to the end of an aluminum rod, whose other end was mounted on the shaft of a closed loop position controlled DC servomotor. The proportional controller applied an oscillatory signal to the target tipped rod at specified frequencies. Incorporated in the feedback loop was a high resolution encoder, which provided an accurate angular displacement measurement independent of the MnScan generated data.

A more detailed description and analysis of this test in which the target was driven at frequencies from 0.5 to 25 hz, and both the target photodetector and the encoder were monitored, is also contained in Sorensen, (1986) (12).

Although further work is necessary, the implication thus far is that this system can record higher frequency dynamic deflections which can be used for the dynamic analysis of manipulators or ultimately for closed loop end-point control.

14. RELATIVE ORIENTATION BETWEEN ROBOT LINKS

One of the motivations in developing this system is to measure the 3-D rotational and translational displacements of one body with respect to another. This information is significant to a number of areas, e.g., tracking the 3-D motion of human joints. It also has been shown that dominant compliance in robots may come from the joints rather than from structural deflection in the links. Thus, the system was used to determine the 3-dimensional joint angles associated with relative motion of two bodies about a moving axis of rotation. These can be obtained based on the premise that three or more points define the position and orientation of a "rigid" robot link.

There are many ways of reducing sets of position data to orientation information, one of which is the SCHUT algorithm (4,10,11). This is a position averaging technique which provides the relative change in position between two bodies. The method used in our system is quite similar, but instead of using position averaging, it averages the orientation to reduce error and to determine changes in orientation. The results from the two methods agree within one degree in most of the cases tested.

To check the accuracy in determining these angles, a single degree-of-freedom jig was designed with each of the two links holding a cluster of four targets fixed with respect to each other. By displacing the jig throughout the measurement volume with the hinged joint angle fixed, the computed orientation angle based on the measured target locations can be compared to the actual angles. This was done for a range of angles for each of the three planes. The results of this test placed the error of the measurement at $\pm$ 0.5 to 1.0 degree at the 95% confidence level. The result represented the accuracy in predicting the relative orientation angle throughout the entire target volume for each of the three planes. Increasing the accuracy in the determination of the XYZ locations of targets, changing the relative locations of the targets in the cluster, and moving the target cluster further from the axis of rotation would improve this result significantly.

Calculation of the "instantaneous" axis of rotation of the joint in question is possible. This measurement can be performed on the system using the reduction algorithms (i.e., SCHUT), but is not entirely stable due to the nature of the computation. This is especially a problem with robot links oriented 180° relative to each other. A new and different method of calculating the orientation between two arms has been developed based on duality theorems and spatial kinematics (8). This method bypasses the instability conditions in other algorithms and thus provides a more accurate result for the rotation angles and the axis of rotation.

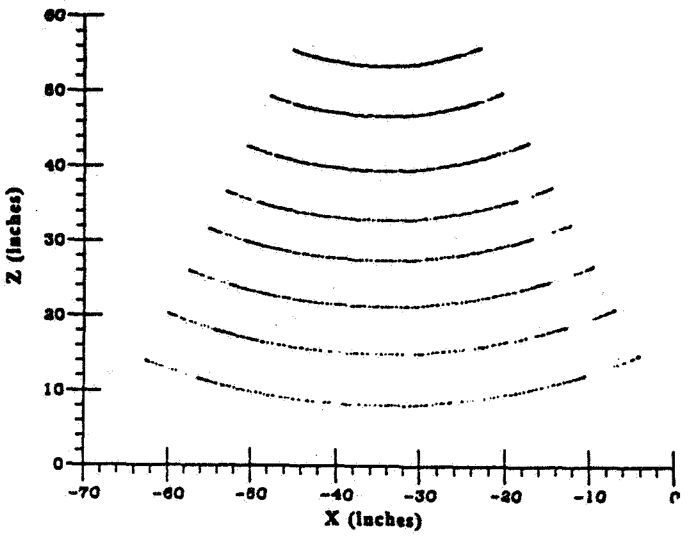


Figure 10a: Cartesian Position Plot of the 120 Hz Case

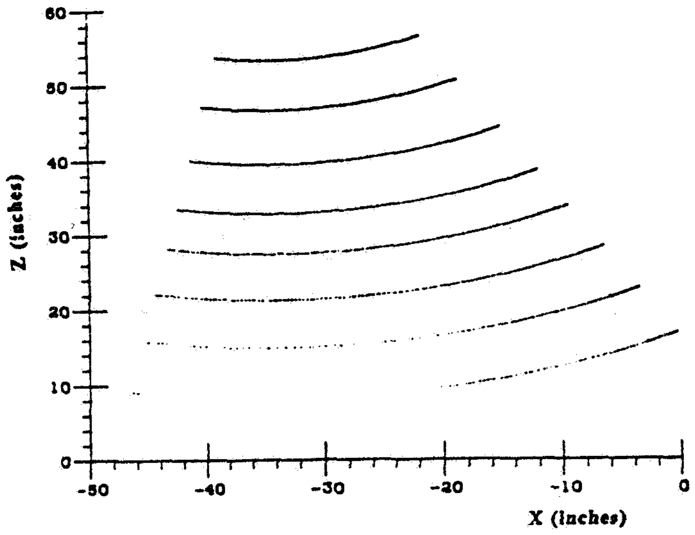


Figure 10b: Cartesian Position Plot of the 480 Hz Case

15. CONCLUSION

The system described provides a fast and accurate method to record human locomotion and end-effector motion parameters. The tests reported here were conducted on the initial prototype pending the imminent completion of a newer system. The earlier prototype is only capable of tracking eight targets; thus data is available only for one joint. The system displays good dynamic bandwidth and acceptable performance. Data collected for 1.75 seconds at 240 Hz for 2 limb segments in a human motion study (4 target per limb) can be reduced to 3-D relative orientation or joint angles on our existing prototype in approximately 20 seconds on an Intel 8086 based system using code generated by a Fortran compiler (with no attempt at optimizing the code). This implies that a computation for a set of instantaneous orientation angles takes approximately 50 msec. A very preliminary test with an array processor has shown that this figure can be cut at least in half.

The accuracy of this system has surpassed the goal specification for human motion tracking for which it was originally designed; our goal had been to design a system with 0.1 inch accuracy. The system is not yet sufficiently accurate for use as an end-point sensor in the control of existing robot manipulators. We, however, do feel that the system can already be used for facilitating quite a number of experiments in this field.

Although the system requires line of sight for operation, the number of light sources and detector targets can be increased to ensure line of sight in most instances. Furthermore, when obstructions to the line of sight are unavoidable, full recovery is achieved as soon as the obstructions are removed.

The resolution of the MnScan system is only a function of the system clock and laser scan speed. The system accuracy at present reflects certain additional phenomena which are being isolated and investigated. Significantly better performance is planned for our next prototype.

16. ACKNOWLEDGMENTS

Partial support for this work was received from the National Institute for Handicapped Research REC Grant No. G008300075 and the National Science Foundation PYI Award (Grant No. DMC-8351827).

17. REFERENCES

1. Bechtold, J. E., "Three-Dimensional Laser Scanning System with Application to Gait and Robotics," M.S. Thesis, University of Minnesota, August, 1983.
2. J. Chen and L. M. Chao, "Positioning Error Analysis for Robot Manipulators with All Rotary Joints," IEEE International Conference on Robotics and Automation, San Francisco, 1986.
3. A. Dainis and M. Juberts, "Accurate Remote Measurement of Robot Trajectory Motion," IEEE International Conference on Robotics and Automation, Saint Louis, 1985.
4. Lenox, J. B., and J. R. Cuzzi, "Accurately Characterizing a Measured Change in Configuration," ASME Paper No. 78-DET-50, 1978.
5. Macfarlane, J. F., "A System for Tracking Motion in Three-Dimensional Space," M.S. Thesis, University of Minnesota, June, 1983.
6. Macfarlane, J. and M. Donath, "Tracking Robot Motion in Three-Dimensional Space: A Laser Scanning Approach," Proceedings of the Nato Advanced Study Institute on Robotics and Artificial Intelligence, Italy, 1983; in M. Brady, L. A. Gerhardt, H. F. Davidson (ed.), Robotics and Artificial Intelligence, Springer Verlag, 1985.
7. Mooring, B. W. and T. J. Pack, "Determination and Specification of Robot Repeatability," IEEE International Conference on Robotics and Automation, San Francisco, 1986.
8. Peterson, S., "Kinematic Analysis of The Human Joint," Ph.D. Thesis, University of Minnesota, Fall, 1985.
9. Schmitz, E. and Cannon, R. H., "Further Experiments in the End-Point Control of a Flexible One Link Robot," to appear in ASME Journal of Dynamic Systems Measurement and Control.
10. Schut, G. H., "On Exact Linear Equations for the Computation of the Rotational Elements of Absolute Orientation," Photogrammetrias, Vol. 17(1):34-37, 1960-61.
11. Schut, G. H., "Formation of Strips for Independent Models," Photogrammetric Engineering, Vol. 34(7):690-695, July 1968.
12. Sorensen, B., "A Laser Scanning System for Tracking Three Dimensional Motion of Human Gait," M.S. Thesis, University of Minnesota, 1986.

J6-63
58.
188126

Object Apprehension Using Vision and Touch

R. Bajcsy and S.A. Stansfield
University of Pennsylvania
Philadelphia, PA 19104

PJ 652116

1. Abstract

We define object apprehension as the determination of the properties of an object and the relationships among these properties. We contrast this with recognition, which goes a step further to attach a label to the object as a whole. Apprehension is fundemental to manipulation. This is true whether the manipulation is being carried out by an autonomous robot or is the result of teleoperation involving sensory feedback. We present an apprehension paradigm using both vision and touch. In this model, we define a representation for object apprehension in terms of a set of primitives and features, along with their relationships. This representation is the mechanism by which the data from the two modalities is combined. It is also the mechanism which drives the apprehension process.

2. Introduction

It has been suggested by both psychologists and perceptual roboticists that objects are defined in terms of their parts and features. It has also been suggested that these features determine not only our recognition of objects, but also our interactions with them. It seems reasonable to say that these features (along with their relationships) are the outputs of the perceptual system. Our first task, then, in building a robotic perceptual system is to determine what this fixed set of features will be. This is also vital to our representation pardigm, since these features must form the building blocks of objects which are to be manipulated by the system. We have chosen a hierarchical representation for objects which consists, at the lowest level, of a set of modality dependent primitives. Examples of such primitives for touch include roughness, compliance, and several types of contact. For vision, primitives are region points and edges. These primitives are extracted by the sensors and then combined into successively more abstract features. During this process, the information from the two modalities is integrated and a symbolic representation is created. For example, a tactile edge and a visual edge are combined into a supermodal entity "edge" which may then be combined with other edges to form a contour. This contour may eventually be labelled as a rim and the rim determined to be part of a pan.

It is interesting to note that each modality measures only some apsect of reality. Take edges for example. The reality of an edge or a boundary of an object is what separates the object from other objects or from the background. Objects here can be either solid, gas or liquid. Sensors, however, only measure only some aspect of this reality. Hence the tactile sensor will measure and detect an edge as the difference between two substances, while the visual sensor will detect an edge as the difference between two colors or brightnesses. This object-background problem is similar to the figure-ground problem of psychophysics. In order to determine what an edge is, we must first determine what the difference is between the object and the background. This leads us to the question of calibration of the background. In our world we assume that the background is air and that the objects are solids. Hence a tactile edge is well defined as the difference between any reactive force from a solid and no force at all (the zero force). Needless to say this is a special, though frequent, case in our universe. A visual edge, on the other hand, does not always correspond to a physical edge. Shadows are an example. Hence the supermodal model must include some control strategies directing the individual modalities to calibrate. In particular, it must calibrate for the object-background relationship. The system might, for example, be required to differentiate between solids and gases in space applications, or between solids and liquids in underwater applications. Without this ability to differentiate between object and background, the concept of an edge is not well defined.

The features and primitives identified by the system are also combined into a hybrid model which is a combination of symbolic and spatial information. This representation is a loosely coupled collection of features and their relations. We call this representation a *spatial polyhedron* and it is, essentially, a user-centered guide to the features of an object and the relationships among these features in space. The identification of the features and relations of an object and their mapping to this spatial polyhedron then constitutes apprehension.

Interestingly, apprehension may be all that is required for grasping: visual apprehension, which gives us global shape and size information, would drive the initial stages of a grasp, such as hand shaping and bringing the manipulator into contact with the object. Tactile apprehension of such features as temperature and roughness would then aid in the fine adjustment stage of the grasp, when information such as weight and smoothness of an object is vital. Our spatial polyhedral representation provides both the visual cues, such as global size, and the tactile cues, such as roughness, which seem to be important to perceptually driven grasping. In addition, it provides information about where the manipulator might expect to encounter each of the parts of an object, both in space (for the initial reach) and in relation to one another (for specific grasps and manipulation).

In the remainder of this paper, we present and discuss the issues involved in designing such a system.

3. System Configuration Issues

In this paper, we discuss the structure of a robotic perceptual system and the integration of information from *different modalities.* Let us begin, then, by presenting a system configuration to serve as a framework for this discussion. What are the issues? First, we consider the type of sensing desired. Sensors are often categorized as either contact or non-contact. Within these two broad classifications, we may place many different types of sensing devices, however, all devices within a classification have several characteristics in common. Contact sensors sense locally -- to gather a large amount of data requires sequential processing of the object. Contact sensors measure surface and material properties directly, for example temperature. And finally, a contact sensor may change its environment. Non-contact sensors tend more often to be global data gatherers, obtaining large amounts of information in parallel. They do not, as a rule, act on their environment (although they may -- for example, a vision system may have its own light source, changable at will.) Given the very different nature of these two types of sensors, it follows that we may make very different use of them. We have chosen to use one sensing mechanism from each of these categories. Our system makes use of a non-contact vison system and a contact sensor composed of a tactile array and a force/torque sensor. When we speak of the perceptual system later in this paper, we will refer to these sensing mechanisms as modalities. This is a term borrowed from the psychology literature. Our reasons for choosing these two sensing devices are twofold. First, they give us different, but complimentary information about the world. And second, they appear to be the two most important senses, in humans, for both recogniton and manipulation.

The next issue which we must consider is how we are to use our devices. Both the visual and the touch systems may be used either actively or passively. Obviously, people use both actively (by which we mean that they are able to control the parameters of the devices at will.) It makes sense that a robot system should also be able to use both modalities actively. However, this is less vital for the vision system, which is able to gather large amounts of data in a single "view", than it is for touch. What is clear, however, is that each sensor is capable of only a partial view of its environment at any one time, and so it is imperative that at least one of the modalities be active.

The final issue is the coupling of the sensing devices. When two devices are coupled, changing the parameters of one will effect the parameters of the other. When they are uncoupled, then each may work independently of the other. One can imagine industrial applications in which the coupling of sensors which represent different modalities would serve the purpose at hand. In the case of general perception, however, it is not clear that the coupling of two modalities will provide any benifits since the information gathered by such devices is conceptually different. What does make sense, though, is to couple two sensors which provide different cues within the same modality -- force/torque and cutaneous, for example. Thus the feedback from one may be used to interpret the information from the other. One can think of various degrees of physical coupling, ranging from rigid coupling (for instance having several sensors on the same probe-finger) through a distributed system coupled via linkages (like the human arm, hand and body), to a physically decoupled system where each sensor system and/or modality functions independently.

The degree of coupling will have important consequences. We postulate that a necessary condition for integrating different sensory systems is that the world being sensed by those sensors which are to be integrated must remain invariant in space during the time interval in which the measurement is taking place or that the system contain some internal knowledge of the nature of the space-time change. We call this invariance spatial-temporal coherence. In the tightly coupled system, where several sensors are positioned on the same probe, the spatial-temporal coherence is guaranted by the physical setup. The disadvantage of this system is that there is no independent control of the data acquisition systems, although there is an independence in the processing of this data (i.e. of the logical sensors). The other extreme case is when the sensory systems are physically decoupled, hence there is independence in the control of the data acqusition process. In this case, in order to be able to integrate the data, and to guarantee the spatial-temporal coherence, one must introduce a supermodal space where the above conditions will be satisfied.

Thus the coupling of sensors will have an effect on both perception and control, particularly during the data acquisition process. This is manifested especially in the haptic modality where different primitives require different hand movement strategies [5]. However, the pairing of primitives and data acquisition strategies (movement of the probe) is universally true as soon as one accepts the concept of an agile (movable) sensor. Take the visual sensor for example, one positions the sensor so that it captures the optimal view and/or detail, depending upon the need. The open question, of course, is the identification of the parameters which will determine what the best view for a given time and context is.

For the remainder of the paper, we will assume that we are dealing with decoupled vision and agile touch sytems, and that the touch system provides us with tightly coupled force/torque and cutaneous sensors.

4. The Building Blocks of Perception

Primitives are the building blocks of perception. They are the lowest level input to the sensory system and require no inferencing capabilities. They are both modality and device dependent. By defining the primitives, we define the features, and hence the objects, which our system will be able to handle. Our first step toward building a perceptual system must therefore be the determination of these primitives. Marr [8], for example, embraces this approach for vision when he presents the successively more complex stages of the visual system begining with zero crossings and ending with surfaces. The primal and two and a half dimenisional sketches embody the features of the system. In machine touch, less work has been done. However, studies with human subjects [5], [7] suggest that the haptic system computes information related to an object's form, substance and function. Form includes measurements of shape and size, while substance represents the properties of an object such as compliance and temperature. While it is not our intention to do cognitive modelling, we feel that the human system provides an excellent example of a working haptic system. Therefore, we propose the following seven primitives for touch: surface normals, contact (edge, point and area), roughness, compliance and elasticity [10]. Temperature, weight and size are also appropriate, but we are not able to detect them with our current device. For vision, the choice of primitives is richer still. For the time being, we will use simply two dimensional region points and three dimensional edge elements.

Once the primitives have been determined, the features of the system may be chosen. Important features for touch are contour, edge, global size and shape, and parts. For vision, surfaces, edges and regions are among the features which may be computed. These features, and their relations, form the output of the perceptual system.

5. Integration Techniques

Given a robot system with multiple sensors, we would like to somehow process and combine the information from each for further use by the system. We refer to this aggregation of disparate sensory data as integration, and it is currently an active topic of research. Several techniques for integration have been explored, each of them taking a very different approach to the problem. Three projects within the Grasp Lab of the University of Pennsylvania illustrate this diversity.

Durrant-Whyte [3] takes a purely mathematical view of the problem. In his research, all sensors are considered as independent agents. The system contains a world model, and the goal is to maintatin the consistency of this model. Objects are modelled as geometric positions using homogeneous transforms, and uncertainty in these positions is modelled as a contaminated gaussian. Integration is achieved mathematically using a baysian statistical model of the sensory data. Resulting changes in the position of the object being sensed are propagated throughout the world model to maintain consistency. There are two aspects of Durrant-Whyte's work as it currently stands which do not make it adequate for perception. The first is that it requires a world model. A perceptual system should make no intial assumptions about the world. The second is that it represents all objects geometrically as homogeneous transforms. Such a representation in not adequate for apprehension or recognition.

Allen [1] applies well-known modelling techniques to the integration problem. The goal is object recognition and objects are modelled geometrically using a Coon's patch representation. Vision and touch are used in a complimentary fashion: Passive vision is first used to define the regions to be explored and to make an initial fit of the data. Touch is then used to explore the regions and to build successively better approximations of the surfaces. In this way, the information from the two modalities is integrated at the level of the geometric model. This instantiation is then matched against a data base of objects created using a CAD/CAM system. Allen's system suffers from the limitations imposed by the use of geometric modelling techniques. It can only recognize precisely modelled objects, although some variation may be allowed by the use of bounds on particular parameters of the object. The recognition of generic objects is not possible.

In our work [9], the goal is object apprehension of generic objects. By apprehension we mean the determination of the properties of an object and the relations among these properties. As we stated earlier, passive vision and one-fingered active touch are used. Objects are modelled symbolically using a hierarchy of frames: frames at the lower levels represent the primitives and features specific to each modality. Intermediate levels represent super-modal features and parts, and at the highest level is a representation of the object as a whole. As the system explores an object, it extracts and identifies the modality dependent primitives and features. Other modules in the system then combine this information into the supermodal entities described above. (As we said in the introduction, this supermodal model must contain the basic physical assumptions about substances (solid, gas and liquid) and the laws that apply to them. These laws, and the subsequent properties which they imply, will then be translated to the individual modalities in terms of expectations (or hypotheses). An important result will be the establishment, for the given world, of the object-background relationship from which the calibration procedure will follow.) Integration within this system occurs at the symbolic level, as modules gather primitives and features (which are themselves already symbolic) and combine them into more abstract entities.

There are several reasons why we have chosen this structure. First, by defining our primitives based upon the sensory systems available, and not upon the objects to be considered, we hope to build a more generalized perceptual system. Because the goal is to apprehend, and eventually recognize, generic objects, it does not make sense to require specific models of each individual object. For the same reasons, we will need to be able to reason about our object categories, both for recognition and for exploration. Reasoning falls into the domain of Artificial Intelligence, and it is from this field that we have chosen to take our representational paradigm. There is also a psychological basis to our design. It has been suggested [11] that humans reason about objects based upon parts and features. Therefore, it seems reasonable to have our perceptual system compute such features and parts.

6. Using Vision to Guide Touch

In manipulation, of which we may consider tactile exploration a subset, there appear to be two stages. First, there is the reach -- a gross motion and orientation mechanism using the arm; then there is the fine adjustment and manipulation stage using only the wrist and fingers. The former is likely feed forward, while the latter uses feedback. It seems reasonable to suggest that the initial reach and hand-shaping is visually-guided, while the fine manipulation (or exploration) is primarily tactile in nature. As a matter of fact, the very properties which each system is most adept at extracting are imperative to the stage at which we suggest it is used. Thus: vision is excellent at determining position in space, rough size and shape, and part segmentation. These are the very parameters required for the initial reach and hand-shaping. Once contact with the object has been made, however, vision may no longer be useful. Often the object is occluded by the grasping hand, or contacting finger. In addition, the parameters important to successful manipulation (again, we use the term to encompass exploration) may not be easily computed by the visual system. One example is the use of roughness and temperature to access the possibility of slip during a grasp. Another is the use of kinesthetic feedback for positioning of the finger during exploration*.

It therefore makes sense to take a "look before you touch" approach, and that is what we have done. The vision system operates first, obtaining initial position, segmentation, and orientation information. This information is used to drive the initial reach and positioning of the finger on the object. The haptic system then takes over to do the tactile exploration. In our case, since we have only a single finger, we choose to approach the object several times and from several different directions in order to fully do the exploration. Because we have no a priori knowledge of the object, and only partial information from our visual system, we need a general exploration method. We use a representation called the spatial polyhedron to accomplish this. The spatial polyhedron is a collection of approach planes. Mapped onto the face of each of these planes is the set of features of the object which one might expect to encounter while exploring the object from that direction. Thus the robot aproaches and contacts an object from each of a set of predetermined orientations. It then invokes the haptic system to explore the features encountered. The end result is a set of extracted features and their relations as defined both implicitly by the relations among the faces of the polyhedron and explicitly by the relations of each feature on a given face. This is in fact apprehension as we have defined it.

7. Some Thoughts About Grasping

We believe that the structure of our perceptual system, and its attendant representations, will extend painlessly to multi-fingered grasping. We have tried to keep the primitives and features dependent only on the modality. Hence they should be as easily computable by several fingers as they are by one. Since the integration within the system occurs at the symbolic level, any number of sensors may input information. New information available only to a multi-fingered hand, such as weight and gross size, can be easily incorporated. Finally, the method by which the reach and object contact are made is designed specifically to be generalizable to a hand, and the spatial polyhedron will allow the simultaneous extraction and aggregation of features from several positions on the object.

Finally, there is evidence that, in humans, grasping and manipulation are perceptually driven, and that the mechanisms for manipulation, such as hand shaping, may actually be part of the stored representation of an object [4]. Thus the development of a haptic perception system, the integration of visual and tactual cues, and the mechanism for visually-guided touch would all appear to be vital to the development of such a perceptually driven manipulation system.

8. Conclusion

In this paper we have presented the framework of a bimodal (contact and non-contact) robotic perceptual system. The concrete study of this general problem is done by investigating vision and touch. Within this framework we have discussed such issues as the system configuration, the choice of perceptual primitives, the integration technique and how vision is used to guide tactile information acquisition. We have further analyzed the consequences of the degree of physical coupling of different sensory systems. We introduce the concept of spatial-temporal coherence and postulate that a necessary condition for integrating different sensory systems is that the world which is being sensed by those sensors which are to be integrated must remain invariant in space during the time interval for which the measurements are taking place or that the system contain some internal knowledge of the nature of the space-time change. Furthermore, the supermodal model must contain facts about the physical world that are true independent of the individual sensors, but that describe the particular world in which the robot must function. This in turn will determine the parameters for calibration of the object-background relationship in the supermodal world, which will then will be translated for the individual modalities.

9. Acknowledgements

The work described herein was performed in the Grasp Lab of the University of Pennsylvania and was supported in part through the following grants: ARO DAA6-29-84-k-0061; AfOSR 82-NM-299; NSF MCS-8219196-CER; NSF MCS 82-07294; AVRO DAAB07-84-K-FO77; NIH 1-RO1-HL-29985-01 DARPA/ONR N0014-85-K-0807.

*A further example of the different ways in which visual and tacile information is processed by the human perceptual system involves the perception of texture [6]. While visual texture is primarly used for grouping and segmentation purposes, the tactile texture determines the properties of a surface, such as roughness. This difference also shows up the data acquisition process: The visual texture detector must be applied over the entire scene, while the tactile texture detector need be applied only locally.

10. References

1. Allen, P. *Object Recognition Using Vision and Touch.* Ph.D. Th., University of Pennsylvania, December 1985.

2. Bajcsy, R., D. Brown, J. Wolfeld, and D. Peters. What Can We Learn From One Finger Experiments? A Report on a Joint US-France NSF-CNRS Workshop on Advanced Automation and Robotics, June, 1982, pp. 119-158.

3. Durrant-Whyte, H. *Integration, Coordination and Control of Multisensor Robot Systems.* Ph.D. Th., University of Pennsylvania, August 1986.

4. Klatzky, R., B. McCloskey, S. Doherty, J. Pellegrino, and T. Smith. Hand Shaping and Object Processing. Cognitive Science - 8605, University of California at Santa Barbara, 1986.

5. Klatzky, R. and S. Lederman. Hand movements: A Window Into Haptic Object Recognition. Paper presented at the 26th Annual meeting of the Psychonomic Society, Boston, Nov. 1985.

6. Lederman, S., G. Thorne, and B. Jones. "The Perception of Texture By Vision and Touch: Multidimensionality and Intersensory Integration". *Journal of Experimental Psychology: Human Perception 12*, 2 (1986), 169-180.

7. Lederman, S. and R. Klatzky . Knowledge-based Control of Human Hand Movements. Paper presented at the conference on Biomechanics and Neural Control of Movement, Henniker, NH, July 1985.

8. Marr, D.. *Vision.* W. H. Freeman and Co., 1982.

9. Stansfield, S. Representation and Control within an Intelligent, Active, Multisensor System for Object Recognition. Proceedings of the IEEE Conference on Systems, Man, and Cybernetics, November, 1985.

10. Stansfield, S. Primitives, Features, and Exploratory Features: Building a Robot Tactile Perception System. Proceedings of the IEEE Conference on Robotics and Automation, April, 1986.

11. Tversky, B. and K. Hemenway. "Objects, Parts, and Categories". *Journal of Experimental Psychology 113*, 2 (June 1984), 169-193.

S7-63

188127

Sensory Substitution for Space Gloves and for Space Robots

P. Bach-y-Rita, J.G. Webster, and W.J. Tompkins
University of Wisconsin
Madison, WI 53706

W4560409

T. Crabb
Astronautics Corporation of America
Madison, WI 53716

A5112015

1. Abstract

~~This paper describes~~ sensory substitution systems for space applications are described. Physical sensors replace missing human receptors and feed information to the interpretive centers of a different sense. The brain is plastic enough so that, with training, the subject localizes the input as if it were received through the missing receptors.

Astronauts have difficulty feeling objects through space suit gloves because of their thickness and because of the 4.3 psi pressure difference. Miniature force sensors on the glove palm drive an electrotactile belt around the waist, thus augmenting the missing tactile sensation.

A proposed teleoperator system with telepresence for a space robot would incorporate teleproprioception and a force sensor/electrotactile belt sensory substitution system for teletouch.

2. Introduction

Sensory substitution is the provision to the brain of information that is usually in one sensory domain (e.g. visual information via the eyes and visual system) by means of the receptors, pathways and brain projection, integrative and interpretative areas of another sensory system (e.g. "visual" information through the skin and somatosensory system). Some examples include sign language for the deaf, Braille for the blind, and the various instrumentation approaches to providing sensory information to persons with specific sensory losses, such as tactile vision substitution systems for blind persons. This paper discusses sensory substitution and sensory augmentation in relation to space needs: augmented sensation for astronauts wearing the bulky gloves required for extravehicular activity and sensory information from space robot hands to the teleoperator.

3. Brain Plasticity as a Basis for Sensory Substitution

Among the most remarkable capabilities of the central nervous system (CNS) is the ability to compensate for losses caused by injuries. This capacity demonstrates that other brain areas are available to assume functions that were previously mediated by the lost neural tissue, or that the functions can be mediated by the remaining neural tissue. This property reflects the plasticity of the brain.

Plasticity is the attribute of the central nervous system in which enduring functional changes take place. It is one of the two fundamental properties of the nervous system; the other is its excitability, which relates to rapid changes leaving no trace in the nervous system.

Sensory information reaches the brain in the form of nerve impulses. There is no doubt that the temporal and spatial patterns of nerve impulses provide the basis of our sensory perception; the coding of information in the form of nerve impulse patterns is a fundamental concept in neurophysiology and psychology. For example, visual information is sent along the optic nerves in the form of patterns of nerve action potentials. The optical images, per se, reach no farther than the retinal receptors. The brain must interpret the nerve impulses as a visual image, after decoding the patterns of afferent impulses. The degree of plasticity available in these mechanisms will determine the functional limitations of sensory substitution systems. In sensory substitution, plasticity is probably the most critical factor of all the properties of the nervous system [1].

The transducer functions of a set of lost or unavailable receptors can be mediated by artificial receptors. For example, in tactile vision substitution systems the TV camera assumes the role. The optical display must be transduced to a form of stimulation that can be handled by the skin receptors, which then assume the functional role of relays. The plastic changes do not occur in the skin receptors or pathways, but in the CNS [1] [2].

4. Some Examples of Sensory Substitution

Two widely used sensory substitution methods are Braille and sign language. The first requires very little instrumentation and the latter, none at all; however, both accomplish the necessary sensory transformation: information usually in one (the lost) sensory domain is transduced to an appropriate display for another, intact sensory system.

(a) In Braille, letters are changed into raised dots; a code based on a 6-dot matrix was developed by Braille to enable blind persons to read letters with the fingertips. The critical factor is that this approach allows the blind person to achieve the same conceptual analysis and mental imagery from reading with the fingertips as the sighted person achieves by reading print.

(b) American sign language (ASL) is an incredibly ambitious and successful sensory substitution system. It translates information usually in the auditory (high frequency, low parallel input) domain into the visual (low frequency, high parallel input) domain. This is accomplished in real time as can be noted by watching a TV news program on which a signer is simultaneously translating into ASL ("translation" is the appropriate term: Bellugi and Klima [3] consider ASL to differ dramatically from English and other spoken languages, with distinct grammatical patterns and its own rules of syntax).

A number of sensory substitution systems requiring high technology have been or are being developed. These include tactile vision substitution, tactile auditory substitution, and tactile somatosensory substitution for insensate hands and feet.

(a) Tactile vision substitution--With a tactile vision substitution system developed in our laboratory, the spatial information gathered by a television camera under the subject's control is delivered to the skin through an array of vibratory stimulators or electrodes. With training, the blind subjects can identify and correctly locate in space complex forms, objects, figures, and faces. Perspective, parallax, size constancy, including looming and zooming and depth cues, are correctly utilized. The subjective localization of the information obtained through the television camera is not on the skin; it is accurately located in the three-dimensional space in front of the camera, whether the skin stimulation matrix is placed on the back, on the abdomen, on the thigh, or changed from one of these body locations to another.

The instrumentation and the research results have been widely reported [1] [4] [5] [6] [7]. A curriculum has been developed to teach congenitally blind children visual spatial concepts, and is being field tested in the United States and Spain.

(b) Tactile auditory substitution--A comparable tactile auditory substitution system has been developed and is now a commercial product (Tacticon). Auditory signals picked up by a microphone are divided into frequency bands and each of these drives one of 16 electrodes on an electrotactile belt worn around the waist, with low tones at one end and high tones at the other [8].

(c) Tactile somatosensory substitution--Some years ago, in collaboration with C. C. Collins, the feasibility of providing tactile information to leprosy patients with insensate hands was explored. A single strain gage was located in each fingertip of a glove worn on one hand, and the information was delivered to the skin of the forehead (where sensation was intact) through five electrotactile stimulators. Within a few hours of training, it was possible to locate the sensation on the fingertips, and it was possible to identify various textures [5]. As with the tactile vision substitution system, correct subjective localization (in this case to the fingertips) required active control of movement by the subject.

The success with the leprosy patient study led to the exploration of other applications. Funded studies are now underway to explore the application of this approach to patients with insensate feet due to diabetes, and to space suit gloves and space robots.

5. Space Suit Glove Requirements

The need for human protection against the space environment began in the Gemini Program of the 1960s. Since then, manned space flight has progressed through the Apollo and Skylab eras, is currently in the Shuttle era, and planning for the Space Station era. Throughout these eras, the space suit glove has evolved into a complicated piece of the extravehicular mobility unit (EMU) and has improved greatly in areas of mobility and dexterity. The EMU is essentially an anthropomorphic enclosure for which the human can operate in the space environment and is shown in Fig. 1 for the Space shuttle era. Extravehicular activity (EVA) has become a much-needed resource in space operations and problems with human performance (in addition to those of a lack of EVA manhours available) are still apparent. In the Space Station era, more than 2000 EVA manhours may be needed to perform construction, assembly, servicing and maintenance activities. Many tasks for the astronauts have been structured around the existing glove capabilities; what activities and tasks could be accomplished with an optimal space suit glove? A perceived optimal glove is shown in Fig. 2.

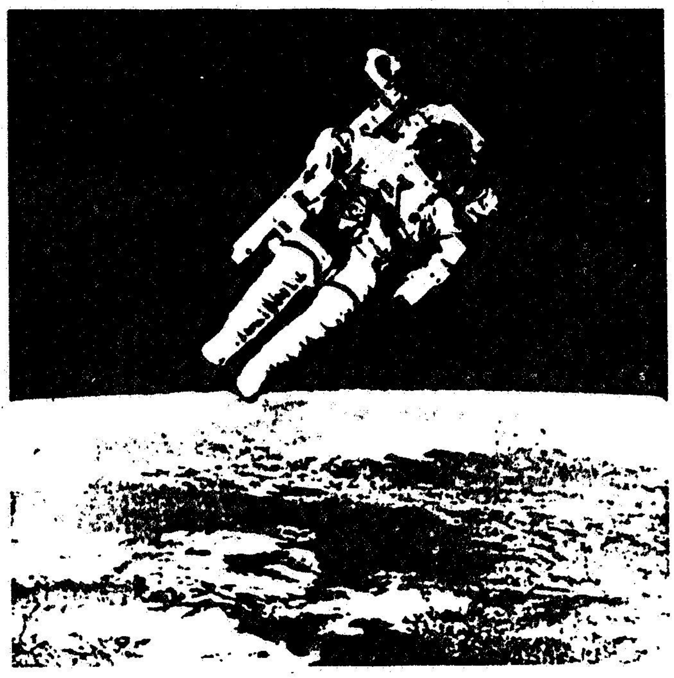


Fig. 1 The extravehicular mobility unit (EMU) is an anthropomorphic enclosure that assists human mobility and dexterity in space.

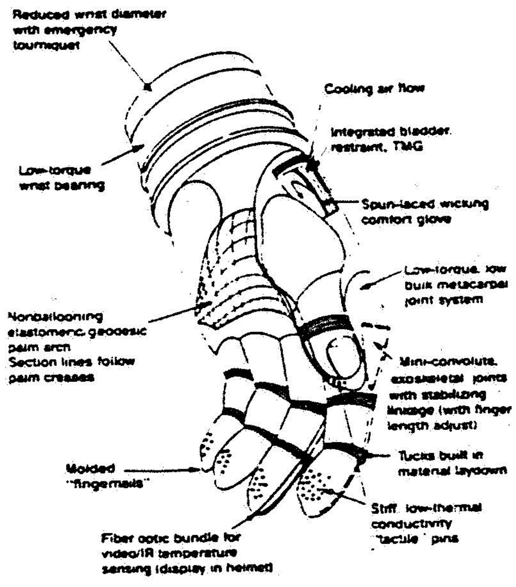

Reduced wrist diameter with emergency tourniquet
Cooling air flow
Integrated bladder, restraint, TMG
Low-torque wrist bearing
Spun-laced wicking comfort glove
Low-torque, low bulk metacarpal joint system
Nonballooning elastomeric geodesic palm arch Section lines follow palm creases
Mini-convolute, exoskeletal joints with stabilizing linkage (with finger length adjust)
Tucks built in material laydown
Molded "fingernails"
Stiff low-thermal conductivity "tactile" pins
Fiber optic bundle for video/IR temperature sensing (display in helmet)

Fig. 2 Optimized EVA Glove.

For the future Space Station, 15 generic EVA activities have been defined as references for EVA system design. These include [9]:

1. Alignment of transmitter and receiver elements
2. Deployment/retraction of solar arrays
3. Truss structure construction
4. Satellite servicing
5. Large module manipulation
6. Small module manipulation
7. Large mirror construction
8. Consumable recharge via module transport
9. Orbit launch operations
10. Satellite operations
11. Space Station radiator construction (from STS)
12. Space Station radiator construction (from Space Station)
13. STS-supported large module manipulation
14. STS-supported truss construction/deployment
15. EVA rescue.

Many of these contain tasks and activities which require fine control and manipulation. Installation of hardware including assembly, replacement of orbital replacement units (ORUs) contingency maintenance and repair, transfer of equipment in and out of pressurized modules, routine support servicing and handling of fluids, equipment stowage, and platform support represent a few of these tasks. The EVA gloves provide the major and sometimes only interface between the astronaut and the work being performed and thus must provide a balance of mobility, tactility comfort, and protection from the workplace hazards.

6. Space Suit Glove Problems

A problem encountered by astronauts during extravehicular activity is that they have trouble feeling objects through their space suit gloves. The glove is made of several layers of plastic and fabric. The plastic prevents air leakage. The cloth provides strength so the plastic will not burst. The cloth also provides thermal insulation and protection from micrometeoroids. It is difficult to feel objects through these layers or through the thick silicone rubber fingertips.

Another even larger problem is that of the astronaut fatigue from work required to move the glove from its neutral position due to the difference between the space suit and space environment pressures. The pressure of the space suit for the Shuttle is 4.3 psi and planned at 8.3 psi for the Space Station. In addition, more radiation and micrometeoroid/debris protection will be required. All of these factors inhibit the design of a flexible and dexterous glove with good sensory capabilities. Answers to both of these problems and many more being addressed by NASA would greatly enhance the performance of the astronaut on EVA. Current training practices have allowed substitution of perception for the lack of tactility. Enhancement of the tactile sensory perception may reduce the fatigue problem, increase the EVA capability to finer motor control, allow the enhancement of the glove design without detriment to the tactility, reduce the astronaut reliance on visual feedback, and thus reduce training time to learn certain tasks. The pressure within the glove causes two major effects. The first is stiffness of the glove itself reducing freedom of position. This restricts the movement of the glove without a great deal of work. Over a large period of time (such as the standard EVA of six hours), the hand is subject to extreme fatigue. Some of this fatigue is due to overgripping to ensure contact. Providing the tactile feedback will give the sense of contact and reduce overgripping thereby reducing the fatigue.

Second, the present air pressure difference of 4.3 psi causes the glove to balloon out. This reduces the tactility between handholds and tools. The resulting tangential forces on the surface of the glove make it difficult to perceive normal forces through the glove. An analogous situation is trying to feel a pebble through a bicycle tire. If the tire is flat, it is easy to feel the pebble through it. If the tire is pressurized, it is very difficult. The proposed space suits would have a 8.3 psi pressure difference, which would make the problem much worse.

The result of this problem is that the astronaut cannot feel when a tool is starting to slip. He overcompensates for this by applying a higher than required force to ensure adequate grasp. His muscles tire quickly and he becomes fatigued much sooner. Force is again used to compensate for this to assure contact. To reduce this ballooning effect, which also reduces the astronaut's capability to grasp an object, NASA has tried hard palms and other palm restraints. The restraint is necessary to enable adequate bending of the metacarpal joint. Solutions to this problem have met with mixed success during the astronaut evaluations. Tactile sensors on the restraint may make such a design feasible in terms of tactility and interfacing with tools, handholds, and other objects.

As Dr. George Nelson reports of his Solar Max Repair activity in space, glove limitations are minimal for gross motor control but are almost inhibitive for fine motor control. His projection indicates that 20% dexterity is lost when handling objects of one inch in diameter and 50% for objects of less than one inch. For objects of millimeter size, the glove permits almost no fine motor control, due to lack of flexibility in conducting more detailed tasks. Dr. Nelson recommends increased tactility especially in the areas of the fingertips and the full length of the index finger and thumb.

Loss of tactility in the space suit glove has been estimated to cause more detriment to EVA performance than is recognized by the crewmembers. Much of the loss of tactility is compensated by visual perception of the task during the many hundreds of hours of training. Providing tactile feedback would relieve the need for such a substitute and possibly reduce the amount of training to perform a specific task.

NASA has looked at mainly passive means in which to increase the tactile feedback to the astronaut including movable pins, enhanced fingertip tactile pads, glove/hand adhesion, and removable fingertips to expose a less bulky, less protected finger. One concept is shown in Fig. 3. Other means such as sensory substitution have also been recommended for feeding other senses such as vision or hearing. The concept discussed here is active tactile sensory perception inducing a simple relocation of the pressure to the abdomen or forearm. Thus the other senses of sight and sound are not overburdened.

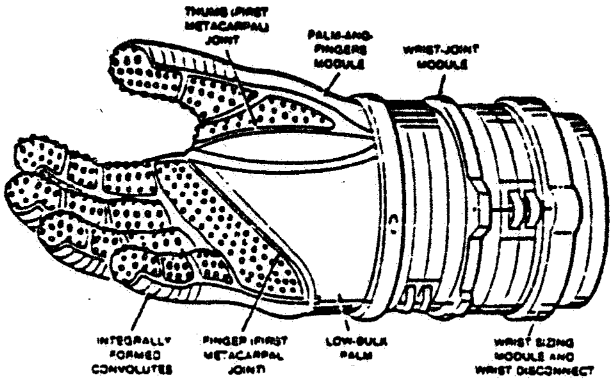

THUMB (FIRST METACARPAL) JOINT
PALM-AND-FINGERS MODULE
WRIST-JOINT MODULE
INTEGRALLY FORMED CONVOLUTES
FINGER (FIRST METACARPAL JOINT)
LOW-BULK PALM
WRIST SIZING MODULE AND WRIST DISCONNECT

Fig. 3 Thermally insulated glove contains short, closely-spaced elastomeric pins that insulate without impairing flexibility.

7. Space Suit Glove Sensory Substitution

Based on sensory substitution mentioned above and reported in detail elsewhere, we proposed a study of tactile sensory substitution for space suit gloves to increase the performance of extravehicular astronauts by increasing tactile sensation. The working hypothesis is that sensory information gathered by the active control of hand movement would be subjectively located in the hand, even though the information arrives at the body at another location (e.g., skin of the abdomen or arm).
We have built a sensory substitution system where the force sensors are on the outside of the glove and are exposed directly to the grasped object. They are located at the points of maximal information, such as the inside of the thumb and first two fingers. We determined these locations by observing wear patterns of used space suit gloves and by grasping various tools using space suit gloves.

A problem in attaching the force transducers is that as the hand is closed, deep wrinkles form on the outer fabric layer of the glove. A small transducer attached to the fabric surface might shift into the pocket of a wrinkle and not detect the surface force at all. To overcome this problem we placed 0.8-mm thick 11-mm diameter metal disks at each location. These had four 1-mm holes at 4 peripheral locations. We used nylon fishline to sew the disks loosely to the fabric. This permitted wrinkles to form but kept the disks parallel to the surface.

Constraints on the selection of a force transducer are that it should be small, have low power consumption, have a wide temperature range, and be rugged. Possible transducer types include strain gages, inductance transducers, optical transducers, piezoelectric transducers, conductive elastomers, time-of-flight transducers, and capacitive transducers. The smallest commercial transducer we found is the Model 105 pressure transducer: 80 psi, 350 ohm from Precision Measurement Company, Box 7676, Ann Arbor, MI 48107. It is 0.28 mm thick and 2.6 mm in diameter. We enclosed the 3 fragile wires from the 1-arm metal strain gage transducer in heat shrink tubing. Using Dow Corning No. 891 Medical Adhesive Silicone Type A, we glued the transducer to the disk and formed a rounded surface of adhesive over the top of the transducer. Thus the rubbery adhesive transmitted forces to the diaphragm of the pressure transducer. We routed wires under the outer fabric layer from the palm to the wrist connector.

We constructed our own electronics to drive a commercial electrotactile system. Because the transducer has only one strain gage element, we added three resistors to complete a Wheatstone bridge. An operational amplifier amplified the small signal from the bridge to a large signal suitable for driving the electrotactile system. Potentiometers to adjust offset and gain were required.

We purchased a Tacticon 1600 electrotactile sensory aid for the deaf. It has a microphone input, divides speech into frequency bands, and delivers electrical stimuli to the skin through 16 gold-plated 5-mm diameter electrodes. A belt around the waist positions the electrodes over the abdomen and receives power from a battery pack and electronics located in a box clipped to the user's normal belt. We removed the speech system and fed the drivers from our amplifiers.

The system worked well in that each user could establish cause and effect between pressure on a specific transducer and electrical stimulation at a particular electrode. Unfortunately our original estimate that 80 psi transducers would be satisfactory was wrong. Firm grasp of tools, such as screwdriver handles overstressed the transducer diaphragms beyond the yield strength, resulting in permanent damage. Thus we are restricted to light pressures. We have ordered 1000 psi transducers and will test those under more rigorous conditions.

We expect that a period of learning will be required for optimal interpretation of this alternative sensory system. We plan an evaluation by skilled astronauts performing typical tasks in a pressurized glove box after an 8-hour learning session.

8. Space Robots

NASA has proposed a space robot that would perform tasks in space under the control of an astronaut in a spacecraft or by ground controllers. The astronaut would control the robot in a master-slave relationship called teleoperation. The astronaut would place his hands and arms in controlling gloves and mechanical arms. When the astronaut would move, the robot would exactly follow him.

Figure 4 shows an additional system, called telepresence, which would provide sensation to the astronaut. When the robot's hand would close on an object, the astronaut's hand inside the controlling glove would feel the same mechanical resistance through proprioception. This teleproprioception [10] would be accomplished by actuators in the astronaut's controlling glove which would be slaved to the actuator in the robot's hand and increase the force that resists controlling glove closure. The astronaut would receive direct position feedback of robot joint angles from his own joint receptors. He would receive direct force feedback of robot forces from his own muscle and tendon receptors. Teleproprioception would provide feedback about the large forces resulting from firm grasp, but not the small forces associated with slip. Figure 5 shows a one-degree-of-freedom system that would provide both teleoperation and teleproprioception.

None of the telerobotic systems described above include teletouch [10]. The human palmar skin contains a variety of mechanoreceptors to sense light touch and nociceptors to sense the pain that results from high pressures. Thus when the robot senses objects at different locations within the palm, this information should be sensed and transmitted to the astronaut. If an object is slipping from the robot's grasp, the astronaut should receive the same sensations as if the object were slipping from his grasp.

Thus the robot's palm should be covered with many force sensors. These should detect small forces necessary to detect slip. They should be capable of withstanding large overloads as when the robot grasps tools firmly. Most transducers have a deformable element such as a spring. Most transducers are designed to operate to an appreciable fraction of their yield strength to give a large output and thus they overload easily. What is needed is an element that deforms easily to produce a sensitive range, but then hits a mechanical stop to be able to withstand high forces during overload. It will be a challenge to develop such transducers in miniature form.

If the astronaut grasped a sharp pointed object, his skin would indent. The ideal teletouch system would also indent his skin. An array of solenoids or air bladders might accomplish this goal, but it would be ~~difficult to miniaturize an array to fit in the controlling glove. If such a system were impossible to~~ develop, the next best solution would be a sensory substitution system. Information from touch sensors on the robot palm would drive stimulators on the palm. These might be vibrotactile arrays located in similar palmar areas within the astronaut's controlling glove. Electrotactile stimulators on the palm are not practical because the thick skin results in painful stimulation. But electrotactile stimulators on a belt around the waist are practical and could be used in a successful system.

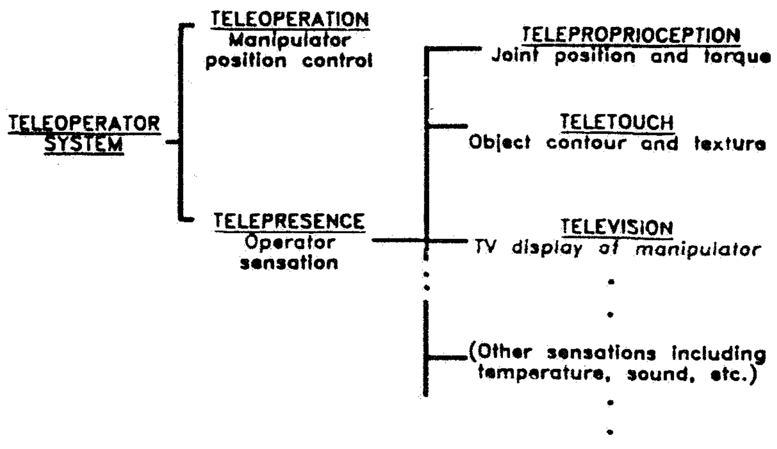

TELEOPERATOR SYSTEM
TELEOPERATION Manipulator position control
TELEPRESENCE Operator sensation
TELEPROPRIOCEPTION Joint position and torque
TELETOUCH Object contour and texture
TELEVISION TV display of manipulator
(Other sensations including temperature, sound, etc.)

Fig. 4 Different elements of a teleoperator system showing the separation of the proprioceptive and tactile sensory feedback.

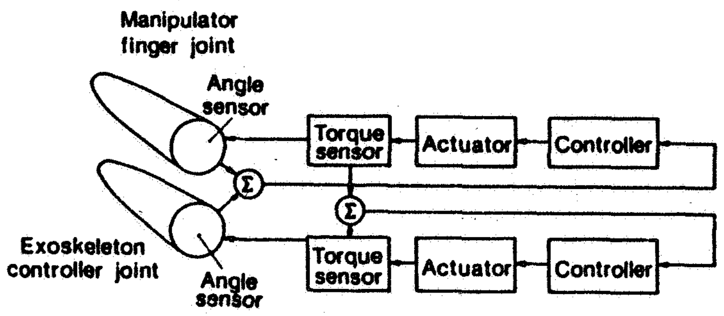


Fig. 5 Block diagram of a one degree-of-freedom force-reflecting control scheme providing teleoperation and teleproprioception.

For a robot to be useful in space, it should be anthropomorphic. Its hand should resemble our hand, so that it can perform the same tasks the astronaut can. Also if the robot fails, an astronaut must perform the robot's tasks. Several groups have developed anthropomorphic-like hands. None has developed teleproprioception for the hand. None has developed teletouch for the hand. It will be a challenge to implement telepresence in the limited space available within the normal boundaries of a hand.

9. Acknowledgements

Work on the space glove was supported by NASA grant number NAS 6 9-171.

Work on the space robot was performed in the Wisconsin Center for Space Automation and Robotics. Support for this center is provided by NASA under grant number NAGW-975 and by a consortium of industrial sponsors.

10. References

[1] P. Bach-y-Rita, Brain Mechanisms in Sensory Substitution, New York, Academic Press, 1972, p. 192.
[2] P. Bach-y-Rita, "Brain Plasticity," in J. Goodgold, ed., Rehabilitation Medicine, New York, C.V. Mosby Co., in press.
[3] U. Bellugi and E. Klima, "Language: Perspectives from another modality," In Brain and Mind, CIBA Foundation Symposia (no. 69), Exerpta Med. Amsterdam, 1979, pp. 99-177.
[4] P. Bach-y-Rita, C.C. Collins, F. Saunders, B. White and L. Scadden. "Vision substitution by tactile image projection," Nature 221, 963-964, 1969.
[5] P. Bach-y-Rita, "Sensory substitution in rehabilitation," in L. Illis., M. Sedgwick, and H. Granville (eds.), Rehabilitation of the Neurological Patient. Oxford, Blackwell Scientific Pub., 1982.
[6] P. Bach-y-Rita, and B. Hughes, "Tactile vision substitution: some instrumentation and perception considerations," in C. Warren and E. Strelow (Eds.): Electronic Spatial Sensing for the Blind. Dordrecht, The Netherlands, Martinus-Nijhoff Pub., 1985.
[7] C.C. Collins, and P. Bach-y-Rita, "Transmission of Pictorial Information through the Skin," Advances in Biological and Medical Physics, 14, 285-315, 1973.
[8] F.A. Saunders, W.A. Hill, and T. Easley, "Development of a Plato-based Curriculum for Tactile Speech Recognition," Journal of Educational Technology Systems, 7, 19-27, 1978.
[9] J.G. Cleland and D.L. Winfield, "NASA Workshop Proceedings: Extravehicular Activity Gloves," Research Triangle Institute, Box 12194, Research Triangle Institute, Box 12194, Research Triangle Park, NC 27709, November 1985.
[10] T.B. Sheridan, "Human Supervisory Central of Robot Systems," in Proceedings of the IEEE International conference on Robotics and Automation, April 7-10, 1986, pp. 808-812.

S8-63
58
188128

Electronic Prototyping

J. Hopcroft
Cornell University
Ithaca, NY 14853

C 5729333

Abstract

The potential benefits of automation in space are significant. The science base needed to support this automation not only will help control costs and reduce lead-time in the earth-based design and construction of space stations, but also will advance the nation's capability for computer design, simulation, testing, and debugging of sophisticated objects electronically.

Progress in automation will require the ability to electronically represent, reason about, and manipulate objects. ~~This paper discusses~~ the development of representations, languages, editors, and model-driven simulation systems to support electronic prototyping. In particular, it identifies areas where basic research is needed before further progress can be made.

Discussed here is

Introduction

An important aspect of automation is the ability to represent, reason about, and manipulate physical objects. This is true in the manipulation of objects in space as well as in the cost-effective design and manufacture of sophisticated parts. A science base is needed to facilitate these activities. This science base must support the modeling and editing of three dimensional objects, electronic prototypes, model-driven simulations, and automated designs. In this paper, the necessary components of the science base are discussed, and a number of examples are presented which illustrate the benefits that would accrue from such research.

Electronic Prototyping

Electronic prototyping is the process of constructing models of physical objects in a computer to support activities such as computer-aided design, engineering analysis, design verification, and automated manufacturing. Electronic prototyping will play an important role in the design, manufacture and operation of space stations as well as having great commercial value.

When compared to the current practice of constructing physical prototypes, the development and use of electronic prototypes (computer models of objects) offer substantial advantages, especially with regard to the design of more complex systems. Take the situation of a satellite with an antenna that deploys in space: it may be the case that the antenna will not support its own weight under gravity and, thus, it would be difficult to construct a prototype from a physical model. An electronic prototype overcomes this difficulty and allows the integration of design and manufacture. A further advantage occurs should the antenna fail to deploy properly. One would be reluctant to experiment with a procedure that might destroy the prototype if only a physical prototype is available for testing hypothesized dejamming procedures. However, an electronic prototype eliminates such worries. Furthermore, one can test hypotheses in parallel on duplicate copies of the antenna model.

A major component of an electronic modeling system is a model-driven simulation. The system must be able to automatically construct the equations of motion from the geometric model. The difference between such a system and current dynamic simulation systems is that during a model-driven simulation a collision detection algorithm is run. Whenever a collision occurs, the dynamics of the system are edited to account for the new point of contact, and the simulation continues. Such a system could be used to simulate gripping and approach strategies for robots, testing designs of multifinger hands, designing algorithms for rotation and manipulation of objects, and studying walking strategies. The ability to redesign a multifingered grip in a few hours and then simulate the new design provides an enormous advantage over building multifingered grippers in hardware. Of course, promising designs must be tested in hardware since simulations often miss pertinent factors.

The versatility of an easily programmed system would allow simulation of a person performing jobs in a weightless environment in space to evaluate various procedures or it could be used to simulate a person on earth allowing us to understand better how tasks are performed. A spinoff of such research would have an impact on many disciplines. For example, when designing artificial joints such as a knee, it is important to understand the forces on the joint as it goes through a range of human activities such as sitting down in a chair, walking up stairs, or stepping off a curb. Knowing the forces would allow verification that the surface of the artificial knee would stay in contact with the surface of the bone for a specific individual with a specific height, weight, and body structure.

One of the first requirements of a modeling system is the ability to construct 3-dimensional computer representations of rigid objects. Although much is known about solid models and boundary representations, very little is known about how to effectively construct or edit a model of a solid. Constructing a model of an object such as an automobile crankshaft from an already existing design may take on the order of a man-month of effort. Clearly, for it to be economically feasible to construct such models, we must produce the tools necessary to significantly reduce this effort. These tools are software tools whose development will require basic research into languages and man-machine interfaces. In order for the cost of constructing the model to be justifiable, the constructed model must support the entire range of engineering activities. These activities include calculations of mass and moments of inertia as well as stress and vibration analyses requiring finite element techniques.

In computer automated design, one observes that a part such as a crankshaft has certain surfaces whose shape is precisely determined by the function of the surface. The shape of other surfaces is not critical and they can be arbitrarily selected provided that they conform to some simple criteria. The crankshaft also has certain global constraints on its design. For example, it must be balanced about certain axes. Thus, one step towards automated design would be to specify the surfaces that need precise shape along with a class of surfaces from which to select the remainder and have the computer complete the design. The crankshaft design would be completed by selecting the remaining surfaces from the class of allowable surfaces so as to satisfy the goal design criteria. An example of the potential savings in time and energy can be seen from [1] where an estimated savings factor of 20 can be achieved in constructing models by automating the blending surfaces.

The required science base for supporting automation is extensive and we can only present a few examples to illustrate the foundational work that must be done. One of the basic areas needing further development is the area of representations. Today there are over fifty commercial solid modelers. Most use boundary representations based on polygonal approximations to the objects, although a number use quadratic surfaces and some use parametric patches. Very little is understood in terms of the trade-offs between the various approaches. For example, using algebraic surfaces requires more computations per face, but using polygonal approximations requires many more faces to represent an object. Whether the increase in the number of faces for the polyhedral approach offsets the savings of the simpler computations is an important question. To resolve it will most likely require the knowledge gained from both the theoretical studies of the type currently taking place in the computational geometry discipline and the practical experience obtained from studies using actual modelers.

Another important avenue that needs to be explored is the development of reliable intersection algorithms for low degree surfaces. To the best of this author's knowledge, no completely reliable algorithm exists for intersecting a quadratic surface with a quartic surface such as a torus. One difficulty in the modeling domain is that degeneracies are the rule rather than the exception. Thus, while many applied mathematicians dismiss ill-conditioned problems with the statement that the problem should be reformulated, the geometric modeler must find efficient and numerically reliable techniques for solving ill-conditioned problems. If two lines l_1 and l_2 intersect a third line l_3 at points sufficiently close together, one can assume that the intersection points either coincide or do not coincide. However, once having made an arbitrary decision, one must insure that at some future point an inconsistent decision is not made. Developing a theory to determine which decisions are independent would be a major accomplishment. Numerous similar questions arise that illustrate the importance of developing a science base to answer such questions in geometric modeling.

Intersection algorithms tend to have running times that are quadratic in terms of the number of faces, since every face must be intersected with every other face. A number of modelers have overcome this by boxing the faces and determining subsets of faces that need not be intersected with other subsets. Empirically, this seems to reduce the execution time of the algorithm from n^2 to $n^{3/2}$. A more promising approach is to find a point on the intersection and trace the curve of intersection to determine the pairs of faces that need to be intersected. The difficulty is that, at the present state of knowledge, the problem of locating two faces that intersect is as hard to solve as the intersection problem itself. In particular, if two objects whose intersection is to be calculated do not intersect, how does the algorithm establish this fact? One promising approach is to triangulate the space exterior to the two objects. This process will either show no intersection or determine a point of intersection. Results in computational geometry suggest that we may soon have techniques to perform triangulations in time order of $n\log n$, a substantial savings over the current n^2 algorithms. More important the $n\log n$ bound is a worst case that is not usually encountered whereas the current n^2 algorithms use time n^2 independent of the intricacies of the problem. Considering that even simple object domains may involve on the order of 10,000 faces, an increase in computing time of a hundredfold is highly likely. Intersection of objects is just one example of an operation on objects. What is needed is the development of the underlying theory to support efficient and reliable algorithms for calculating Boolean operations, swept volumes, offsets, envelops, triangulations, etc.

Another major area that needs investigation is that of editing objects. Today there are good editors for text and good editors for programs because we understand the structure of text and the structure of programs. Text consists of paragraphs that consist of sentences, sentences are made up of words, and so on. A good editor makes use of this structure. In programming, good programmers do not start from scratch each time they construct a new program. Rather, they select a previous program in one window and use bits and pieces to construct a new program in another window. It is essential that in modeling components we develop the ability to reuse pieces of old models. So far this has not happened. A better understanding of the internal structure of physical objects needs to be achieved. For example, consider a cube defined as the intersection of three slices, an x-slice, a y-slice, and a z-slice. Let the planes defining the x-slice be called left and right, the planes defining the y-slice be called front and back, and the planes defining the z-slice be called the top and bottom. Let the front_right_top vertex be the intersection of the front, right, and top planes. Now consider graphically editing the cube by moving the front_right_top vertex. Once the vertex is moved the three planes must be moved to maintain the constraint that the front_right_top vertex is the intersection of the front, right and top planes. In this case the cuboid changes in dimensions but is still a cuboid. On the other hand if the cuboid had been defined with the vertices as basic elements, the edges as lines connecting certain pairs of vertices, and the faces as being patches, then moving the front_right_top vertex has a quite different effect. In particular, the coordinates of the vertex are modified, three edges incident at the vertex change their orientation, and three faces change from being planar to hyperbolic paraboloids. The cuboid ceases to be a polyhedral figure and has curved faces.

This example illustrates that geometry alone does not capture the nature of a physical object. In fact, it may well be that for editing purposes an abstraction of the object devoid of geometry is essential. The editing and reuse of the design is done at the level of the abstraction, and the geometric properties are then derived from the abstraction. Ultimately, as models are created, it is clear that some geometric properties will have to be symbolically recorded. For example, the threads on a screw do not need to be geometrically represented for most applications. However, they must somehow be represented. As important as the reuse of previous designs is, surprisingly little research has been devoted to this area; a science base is almost completely lacking. This is likely to seriously impede the nation's efforts to automate the design and manufacturing process and will critically affect areas of high technology such as space exploration.

User interfaces will play an important role in increasing productivity. To illustrate the effectiveness of well designed user interfaces, consider the example of initializing a simulation of a man diving off a block. To describe to another person how to place the diver on the block, one would simply say "place the diver on the diving block". The other person would automatically understand that the diver should be placed with feet on the block, standing in an upright position, and facing forward with hands at sides. The fact that humans have internal models of the world allows them to communicate complex situations to others using relatively short messages. The fact that one human knows by and large how another human will interpret information allows him or her to structure communications so that the correct interpretation will be achieved. Computer interfaces are needed that allow the same ease of communication.

With our current software interfaces, describing the initial position of the diver would be a tedious task. Looking at a very simplistic model, assume the diver has one hundred degrees of freedom. In this instance, the user would need to specify one hundred parameters. Although well designed systems have default specifications, it is highly unlikely that default specifications would greatly reduce the number of parameters needed for the diver. However, a simple algorithm that takes advantage of partially supplied knowledge to fill in defaults might make human-to-computer communications almost as effective as communication between humans. For example, if points A and B on an object were specified to be placed at A' and B' in space, the algorithm might fix the remaining degrees of freedom by translating A to A' and then performing a minimum rotation to get B to B', i.e., a rotation in the plane determined by the points A, B and B'. There would be no extraneous rotation about the AB axis. The reason a set of simple heuristics such as the above is powerful is that the user understands how defaults are supplied and quickly learns how to initialize objects with minimal information. A simulation system with an easy interface for describing models, tasks, and initial configurations would be a powerful tool for developing such things as a robot arm capable of manipulating and repairing satellites in space. The design of such an arm could be greatly enhanced if one could easily edit a design to try out various ideas and easily specify procedures for using the arm.

A very promising avenue of research is symbolic specification. Objects can be assembled and manipulated symbolically by developing automatic naming conventions and inheritance methods. A human has considerable difficulty with coordinate systems in 3-space. Thus, rather than trying to specify location directly, locations are specified by constraining a feature of one object to mesh with a feature of another object. This usually requires the computer to maintain and solve systems of constraints. Research is needed in this area to eliminate the need to solve arbitrarily complex systems of constraints and to rapidly detect inconsistent systems of constraints.

So far we have talked almost exclusively of objects that are physical entities. In addition to thinking about physical objects, we must also think of things such as tasks, trajectories, etc. and understand how to represent and edit them. In using a robot to perform several similar tasks, it would be preferable to take the code that was developed for one task, edit it, and reuse it for other tasks. This may be tedious to do at the code level since minute differences in the tasks may cause considerable differences throughout the code. Representations of the tasks at some level of abstraction from which code could be automatically produced to drive the robot would allow simple editing and allow the conversion from one task to another. The idea is a simple extension of the concept of high level programming; editing the source in the high level language is enormously easier than editing the machine language object code.

In the above we have tried to illustrate the need for a much better understanding of the software representations of objects and tasks. In addition, numerous aspects such as motion planning and configuration spaces, constraint systems, and symbolic computations involving ideals and the Grobner basis need a better understanding. Although robotics and automation deal with physical objects and are often thought of in terms of control, sensing, and instrumentation, the real nature of the subject has to do with representations, languages, abstraction, and reasoning. While these are generally computer science concepts, the researchers in robotics tend to have backgrounds in electrical and mechanical engineering. There is a need to integrate computer science into these fields. The newness of these ideas and the lack of sufficient researchers with training in computer science has contributed to the slow development of these areas.

It is crucial that the nation build the science base to support automation. The greatest challenge will be the development of the foundations in representations, languages, and user interfaces for the computing systems involved. A well thought out approach in this area is strongly needed.

Reference 1. Hoffmann, C.M., and Hopcroft, J.E. Automatic Surface Generation in Computer Aided Design, *The Visual Computer 1:2*, 1985, 92-100, Springer-Verlag.

S9-63
68
188129

Multiple Degree of Freedom Optical Pattern Recognition

D. Casasent
Carnegie Mellon University
Pittsburgh, PA 15213

CH 188052

ABSTRACT

Three general optical approaches to multiple degree of freedom object pattern recognition (where no stable object rest position exists) are advanced. These techniques include: feature extraction, correlation, and artificial intelligence. The details of the various processors are advanced together with initial results.

1. INTRODUCTION

This paper addresses object pattern recognition for *multiple degree of freedom* (M-DOF) image cases. This is defined as the recognition and identification of an object with no stable rest position. We emphasize *optical pattern recognition* (OPR) techniques and research for this problem, with recent results obtained at the *Center for Excellence in Optical Data Processing* at Carnegie Mellon University. Three different optical processing techniques are addressed and highlighted. These include: feature extraction (Section 2), correlation (Section 3) and optical artificial intelligence (Section 4).

2. OPTICAL FEATURE EXTRACTION FOR M-DOF PATTERN RECOGNITION

The general feature extraction approach to pattern recognition [1] is diagramed in Figure 1. In this section, we emphasize different feature spaces that can be optically realized. The feature extraction and classification techniques are established [2]. All feature spaces we consider are in-plane distortion-invariant. We achieve 3-D M-DOF distortion-invariance by training sets and use of *linear discriminant functions* (LDFs).

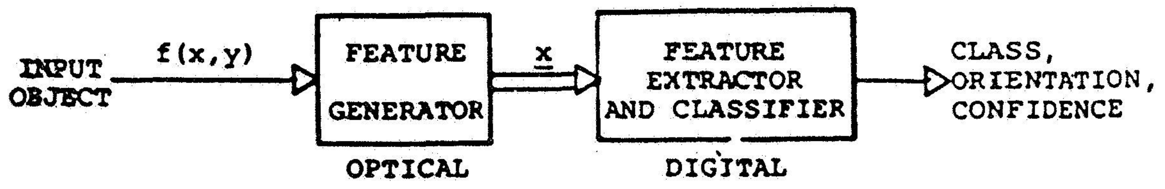

INPUT OBJECT
f(x,y)
FEATURE GENERATOR
OPTICAL
x
FEATURE EXTRACTOR AND CLASSIFIER
DIGITAL
CLASS, ORIENTATION, CONFIDENCE

Figure 1: *Hybrid Optical/Digital Feature Extraction Processor Block Diagram*

2.1 CHORD DISTRIBUTION FEATURE SPACE

This feature space consists of the distributions h(r) of the length (r) and the distributions h(θ) of the angles (θ) of all chords associated with an input object. We

PAGE 64 INTENTIONALLY BLANK

allow gray-level objects, internal object points, and all chords associated with these input object points (if the internal object points are reliable) in our synthesis algorithm. We achieve generation of this feature space [3-4] with the system block diagram in Figure 2. This feature space provides in-plane distortion-invariance. We achieve out-of-plane distortion-invariance by the use of training set data and LDFs. The case studies for which this feature space has been tested included a set of ship data and a set of aircraft imagery [3,4]. The LDFs used were Fisher vectors and dominant Karhuhen-Loeve eigenvectors. Most attractive results ($\simeq$ 95% correct classification) were obtained.

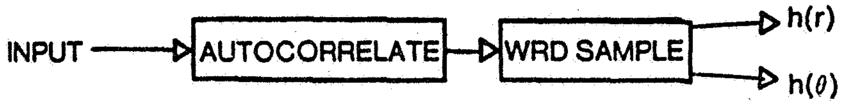
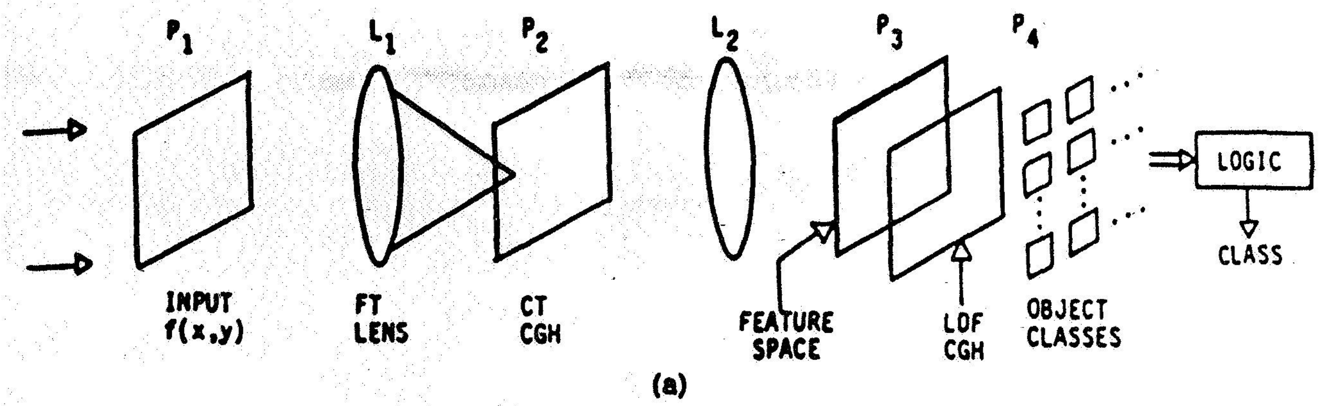


Figure 2: *Optical Chord Transform Feature Space Generation Block Diagram*

2.2 SPACE-VARIANT FEATURE SPACE

An attractive feature space that is in-plane distortion-invariant can be obtained from the Fourier transform of coordinate transformed in-plane data [5]. The resultant system has a different impulse response at each spatial point in the system. The coordinate transform is chosen to make the features invariant to different geometrical distortions. A polar transform results in rotation invariance. If the logarithm of the axes is taken, the features are scale invariant and a Mellin transform results. If we log the radial axis in polar space, scale and rotation invariance are both achieved. To obtain shift-invariance, the object must be centered (by moments, etc.) or the coordinate transform operations can be performed on the magnitude Fourier transform of the input data. Figure 3a shows an optical system to achieve this. The *coordinate transformation* (CT) is performed by a *computer generated hologram* (CGH) at P_2. The output feature space at P_3 can be operated on in parallel by optical LDFs implemented on another CGH. In this case, the class of the input object is determined by the location of a peak in P_4 on a particular detector, or by the binary-encoded output value from a set of N detectors. Figure 3b shows the block diagram of this space-variant processor [6].

As a demonstration of the use of this architecture for M-DOF object identification, we consider a set of 9 different aircraft. These objects have no stable rest position and thus represent an attractive application for an M-DOF processor. Since the feature space at P_3 is in-plane (scale, rotation and translation) invariant, we use a training set to provide out-of-plane invariance (in pitch and roll of the aircraft). A relational graph was devised to identify the class of the aircraft. At the first level of the graph, a decision is made on the sub-class of the object (e.g. commercial, fighter, etc.). A *synthetic discriminant function* (SDF) LDF was used at this node for this decision. At subsequent nodes, the name class (F104, DC10, etc.) of the aircraft is determined. This represents a multi-class graph (with greater than one decision, i.e. one of three choices, made per node). A second binary graph (with one of only two decisions made per node) was then devised using Fisher LDFs. In both graphs, different features (the



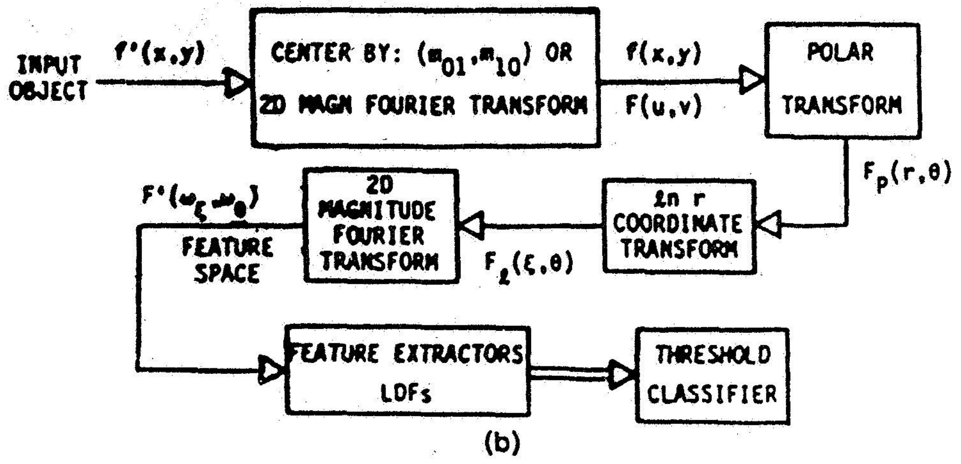


Figure 3: *Optical Space-Variant Feature Space. (a) Optical System; (b) Block Diagram*

optimal ones) are used at different nodes. The training set consists of 5 images per object class (aircraft name class) at 0° and ±20° rotations in pitch and roll (recall that the feature space is invariant to yaw, as well as scale differences). The graphs were then tested versus 0°, ±10°, ±20° and ±30° in pitch and roll. (These are distortions for which the feature space is not automatically invariant. In other tests on in-plane distortions, all results were positive and thus are not included in these M-DOF tests.) The full test set thus consisted of 13 images for 9 different aircraft (117 images). The results obtained were approximately 99% and 95% correct recognition for the two graphs. This demonstrates the M-DOF performance of this feature space processor.

2.3 MOMENT FEATURE SPACE

A moment-based system (block diagramed in Figure 4) has also achieved M-DOF recognition [7,8]. In this system, the moments are optically generated. The first level classifier uses the ratio μ_{20}/μ_{02} to estimate the aspect of the object and a hierarchical tree to estimate the object class. The results from these first-level estimators are used to access 21 moments for each object class. These are then used in an iterative second-level estimator to confirm the object class, its distortion parameters and the confidence of the estimates. This M-DOF processor has been successfully tested on data bases of pipe parts [7] and ship data [8].

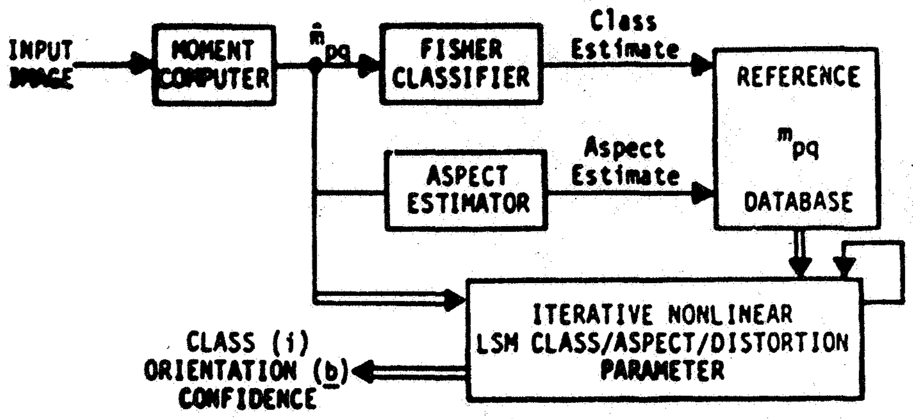


FIGURE 4: ***M-DOF Moment-Based Pattern Recognition System Block Diagram***

3. M-DOF OPTICAL CORRELATORS

Optical correlators represent one of the most powerful optical systems. They provide shift-invariant recognition of multiple objects in parallel in the presence of high- clutter. With space-multiplexed filters, one can correlate an input scene versus several filter functions in parallel and either produce multiple output correlation planes or superimposed multiple correlation plane outputs. Frequency-multiplexed filters also enable multiple output correlation planes. One can employ frequency-multiplexed filters at each spatial-multiplexed filter location. With *holographic optical elements* (HOEs) lenses on each filter, various summations of multiple output correlation planes are possible. These architectures are limited in practice by the number of 2-D correlation planes one can read out in parallel and by the lack of distortion-invariance in correlation *matched spatial filters* (MSFs). We now discuss distortion-invariant MSFs, a hierarchical correlator and a symbolic correlator for M-DOF processing.

3.1 DISTORTION-INVARIANT FILTERS

We have devised various techniques to synthesize distortion-invariant correlation filters from training set data [9,10]. These are referred to as SDFs. We can specify the peak value of the correlation output in most of these filters. The three types of filters are: projection filters (these specify only the correlation peak value), output correlation filters (these specify the shape of the correlation function), and *peak to sidelobe ratio* (PSR) filters (these maximize PSR, but cannot control the correlation peak value). These filters have been synthesized to recognize an object independent of its aspect view. Initial tests have been most successful for ATR, ship and aircraft targets.

3.2 HIERARCHICAL CORRELATORS

These distortion-invariant filters allow one filter to recognize an object independent of distortions. They thus significantly reduce the number of filters necessary and hence correlation planes to be analyzed. The use of K multiple filters with binary encoding of the outputs enables K filters to recognize 2^K object classes. Control of the filter peak outputs to L levels allows F filters to handle L^F object classes. Thus, these filters allow large object class problems with a small number of filters and with the other advantages of a correlator. In extensive tests, we find that as the size of the problem to be solved increases, the filter's performance degrades. A proposed solution to this is a hierarchical correlator [11]. In the first level of this system, PSR filters are used to locate *regions of interest* (ROIs) or candidate objects in the scene. Correlation filters are then used in the second level to test each location and the shape of the correlation peak there. In the *final* level of the hierarchical system, projection filters are used to confirm the object class and to identify it and to determine its orientation.

3.3 SYMBOLIC CORRELATORS

One can view correlation outputs from multiple filters as a symbolic description of the input object. The use of multiple multiplexed filters with symbolic post-processing offers significant potential for M-DOF multiple object pattern recognition in parallel.

4. OPTICAL ARTIFICIAL INTELLIGENCE

Various optical AI processors have recently emerged and have been advanced. Initial remarks on each are now advanced. More extensive tests on all are necessary to more fully assess each. The relational graph processor in Figure 3 is one approach. The automatic organization of data into sub-classes as employed in this processor is a useful technique for any knowledge base or inference system. A model-based description of objects is another approach that is most attractive because of its efficient storage and its ability to easily generate different object aspect views. A *reference function generator* using this concept is quite general purpose and useful for filter synthesis and generation of the filters for correlators and for the memory matrices in associative processors. Successful initial tests on aircraft data has been most attractive using these approaches. We expect future work to concentrate on optical AI techniques, hopefully with attention to system realization and to more extensive testing.

5. SUMMARY AND CONCLUSION

Figure 5 shows one version of three levels of the hierarchical vision processing system, the scene and object elements involved in each and the type of processing employed at each level. As seen and as was briefly described above, there is a significant role for optics in each level of vision.

ACKNOWLEDGMENTS

The support of future work in this area by a NASA JPL contract is gratefully acknowledged.

REFERENCES

1. D. Casasent, "Hybrid Optical/Digital Image Pattern Recognition: A Review", Proc. SPIE, Vol. 528, pp. 64-82, January 1985.
2. R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis, Wiley & Sons, New York, 1973.
3. D. Casasent and W.T. Chang, "Generalized Chord Transformation for

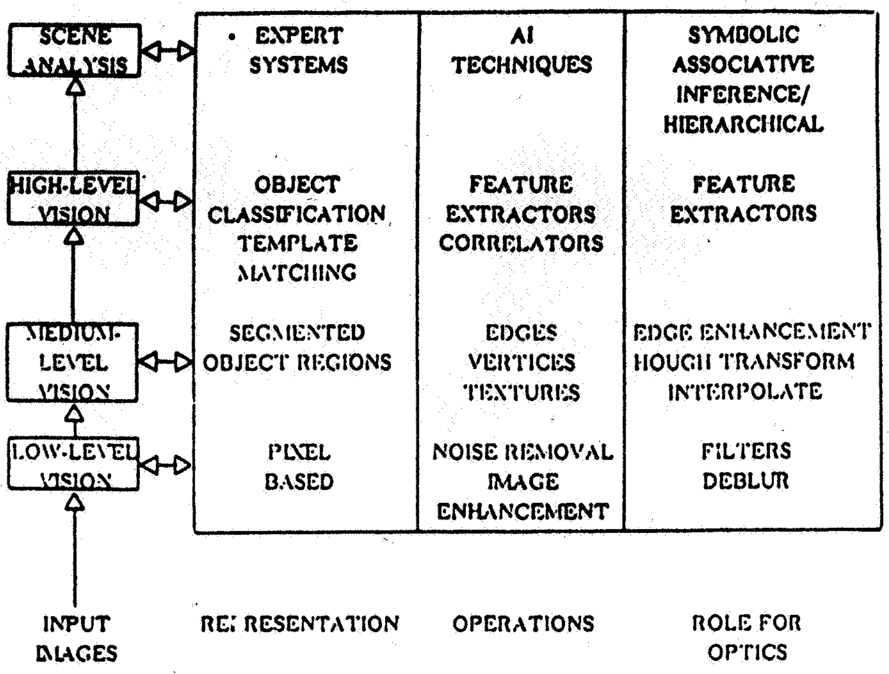


Figure 5: *Hierarchical Levels of Vision, Scene, and Object Recognition and the Role for Optics in Each*

Distortion-Invariant Optical Pattern Recognition", Applied Optics, Vol. 22, pp. 2087-2094, July 1983.

4. D. Casasent and W.T. Chang, "Parameter Estimation and In-Plane Distortion Invariant Chord Processing", Proc. SPIE, Vol. 579, pp. 2-10, September 1985.
5. D. Casasent and D. Psaltis, "Deformation-Invariant, Space-Variant Optical Pattern Recognition", Chapter in Progress in Optics, E. Wolf Ed., Vol. XVI, Holland Pub. Co., New York, 1978, pp. 291-356.
6. D. Casasent and A.J. Lee, "An Optical Relational-Graph Rule-Based Processor for Structural-Attribute Knowledge Bases", Applied Optics, Vol. 15, , pp. 3065-3070, 15 September 1986.
7. D. Casasent and R.L. Cheatham, "Hierarchical Pattern Recognition Using Parallel Feature Extraction", Proc. ASME, *Computers in Engineering 1984*, Vol. 1, pp. 1-6, August 1984.
8. R.L. Cheatham and D. Casasent, "Hierarchical Fisher and Moment-Based Pattern Recognition", Proc. SPIE, Vol. 504, pp. 19-26, August 1984.
9. D. Casasent, "Unified Synthetic Discriminant Function Computational Formulation", Applied Optics, Vol. 23, pp. 1620-1627, May 1984.
10. D. Casasent and W.T. Chang, "Correlation Synthetic Discriminant Functions", Applied Optics, Vol. 25, pp. 2343-2350, 15 July 1986.
11. D. Casasent, "Optical AI Symbolic Correlators: Architecture and Filter Considerations", Proc. SPIE, Vol. 625, pp. 220-225, January 1986.

S10-63
188130
13P.
188130

Maximum Likelihood Estimation of Parameterized 3-D Surfaces Using a Moving Camera

Y. Hung, B. Cernuschi-Frias, and D.B. Cooper
Brown University
Providence, RI 02912

B172031

(the present case)

Abstract

A new approach is introduced to estimating object surfaces in three-dimensional space from a sequence of images. A surface of interest here is modeled as a 3-D function known up to the values of a few parameters. The approach will work with any parameterization. However, in work to date we have modeled objects as patches of spheres, cylinders, and planes,---primitive objects. These primitive surfaces are special cases of 3-D quadric surfaces. Primitive surface estimation is treated as the general problem of **maximum likelihood parameter estimation** based on two or more functionally related data sets. In ~~our case~~, these data sets constitute a sequence of images taken at different locations and orientations. A simple geometric explanation is given for the estimation algorithm. Though various techniques can be used to implement this nonlinear estimation, we discuss the use of gradient descent. Experiments are run and discussed for the case of a sphere of unknown location. These experiments graphically illustrate the various advantages of using as many images as possible in the estimation and of distributing camera positions from first to last over as large a baseline as possible. In order to extract all the usable information from the sequence of images, all the images should be available simultaneously for the parameter estimation. We introduce the use of asymptotic Bayesian approximations in order to summarize the useful information in a sequence of images, thereby drastically reducing both the storage and amount of processing required. The attractiveness of our Bayesian approach is that now all the usual tools of statistical signal analysis can be brought to bear, the information extraction appears to be robust and computationally reasonable, the concepts are geometric and simple, and essentially optimal accuracy should result.

[illegible]

I. Introduction

Essentially all 3-D object surface estimation from multiple views to date is based on either active stereo using a laser and one or two cameras for triangulation, or on passive stereo involving matching points in two images and using triangulation, or on optical flow [1], [10], [11]. We suggest a new approach in which surfaces of complex objects are approximated by a few patches of 3-D parameterized surfaces, and these parameters are estimated from two or more images taken by calibrated cameras from different locations and directions. These parameterized patches are referred to as *primitive objects*. We formulate the parameter estimation problem as standard maximum likelihood estimation, given two or more functionally related data sets. Estimation accuracy is achieved by processing data in blocks (which may be large), in addition to processing many images and with camera positions distributed over as large a baseline as possible. The actual processing is simple standard statistical signal analysis. This approach, first presented in [4], is completely new as far as we know. In summary, the contribution of this paper is the treatment of 3-D surface inference as a standard *maximum likelihood parameter estimation problem* requiring low data storage capacity and where parameter estimates are updated recursively as each new image in a sequence of images is received and processed.

Central to 3-D surface estimation from two (or more) images taken from cameras in different locations and orientations is the pairing of points from two images that are images of the same point on a 3-D surface. This matching of points in two images is usually done in either of two ways. (i) If the two cameras are physically close and their optical axes are almost parallel, then their images will differ from one another only by translation---one will be a *shifted version* of the other. Then image 1 can be partitioned into patches, and each patch cross-correlated with image 2 to find its location in image 2. Once this correspondence is known, the location of the surface region in 3-D space seen in the pair of corresponding image patches can be determined by triangulation. Since the surface region seen is usually curved, one would like the patches to be small in order to locate the surface region seen accurately. However, if the images are noisy, large surface patches must be used to accurately estimate a pair of corresponding patches in the two images.

Significant triangulation errors occur when the camera optical axes are close together and almost parallel because of matching errors due to image noise, and because 3-D object surfaces are curved. Additional triangulation error occurs because there is some error in camera calibration. (ii) An alternative approach that permits a large angle between the camera optical axes to improve triangulation accuracy is to locate corresponding small local features in the two images. An example of such a feature is a vertex of a polyhedron. For a curved surface, contours on the surface are features often used to be matched in pairs of images. The difficulty here is that a large amount of pattern recognition may be necessary to recognize a pair of corresponding features in the two images. Past efforts at cross-correlation of large image patches, as in (i), has been unsuccessful here because a patch in one image will be a distorted version of a corresponding patch in the other image.

The work closest in spirit to ours is the recent work of Faugeras, Ayache and Faverjon [8], who develop the idea of estimating points and lines on a 3-D object surface, or planar surfaces, from a sequence of images. More specifically, they assume that the probability distribution for the estimates of points on a surface based on a pair of images is known. They then assume that a sequence of such estimates and associated distributions are known for a sequence of images. Their contribution, then, is to use the extended Kalman filter for combining this sequence of estimates to obtain improved estimates of the surface points. They derive the equations for estimating lines, and suggest that it can be extended to planes. Among the errors they take into account, are those in camera calibration. Their concept is important, though they do not tackle here the problem of optimally estimating the surface points or lines directly from the data in the images.

Our paper is an expansion of one where our 3-D surface estimation algorithm was first proposed [4]. In subsequent papers, we showed that our basic estimation algorithm is maximum likelihood estimation, and derived Cramer-Rao *irreducible* lower bounds on the parameter estimation error covariance matrices [6], and we also discussed the use of Markov Random Field (stochastic process) models for 3-D surfaces [5] as a generalization of the use of parameterized surface models. These and the present paper together constitute a new Bayesian theory for 3-D surface estimation based on a sequence of noisy images.

Sections II.A - II.C introduce the transformations necessary for understanding the relation of images in two or more views. Sections III.A - III.B describe the performance functional and the gradient descent algorithm used in estimating the a priori unknown 3-D object parameters based on the use of two images. Section III.C provides a very simple geometric interpretation of the algorithm. Sections IV.A - IV.D extend the approach for use of a sequence of images that might be taken by a moving camera. In order to arrive at a computationally feasible algorithm, we introduce the use of maximum likelihood estimation here. This development also points out that the algorithm described in section III is maximum likelihood estimation. The importance of this observations is that maximum likelihood estimators are known to converge to the true parameter values, and are known to have minimum estimation error covariance as the number of observations become large. In section V we introduce a somewhat different estimator for a moving camera, and point out that it has certain desirable computational properties but is *less accurate*. This algorithm is somewhat similar to the use of *optical flow*.

II.A Notation and Description of Camera Motion

Let P be a point in 3-D space and $\mathbf{r} = (x \ \ y \ \ z)^{T}$† *be its coordinates in the fixed orthogonal world reference frame*. Since we assume that objects do not move, this reference frame is fixed with respect to the objects viewed by the camera, and we will call it the object reference frame (ORF). Let $\mathbf{r}(n) = (x_n \ \ y_n \ \ z_n)^T$ be the coordinates of the point P in CRFn, the reference frame attached to camera n. This reference frame is such that: (1) the camera optical axis is parallel to the z_n axis, and it looks at the negative z_n axis; (2) the x_n and y_n axes are parallel to the sides of the image; (3) the origin of the reference frame coincides with the center of the image plane. The image is corrected so that the view is not inverted top to bottom and left to right, i.e., a central projection is used.

Let B(n) denote the 3×3 orthogonal rotation matrix that specifies the three unit coordinate vectors for CRFn in terms of the three unit coordinate vectors for the ORF. Let $\mathbf{r}_c(n)$ specify the origin of CRFn in the ORF. Then

$$\mathbf{r}(n) = B^T(n)\left[\mathbf{r} - \mathbf{r}_c(n)\right], \qquad \text{and} \qquad \mathbf{r} = B(n)\mathbf{r}(n) + \mathbf{r}_c(n). \tag{1}$$

The rotation matrix B(n) and the translation vector $\mathbf{r}_c(n)$ are known for calibrated cameras. In this paper, we will use $\mathbf{b}_n$ to represent a vector having as its components the parameters that specify both B(n) and $\mathbf{r}_c(n)$.

† A symbol in boldface is a column vector, a superscript capital T attached to a vector denotes vector transpose.

II.B Surface Parameterization

Our approach is applicable to *any parameterized surface*. A few researchers have used differential geometric properties, such as Gaussian curvature and mean curvature, to describe surfaces, see [2]. These are useful for surface parameterization because they are coordinate free. In general, the surfaces we want to estimate can be described by an implicit function with respect to the ORF:

$$g(\mathbf{r};\mathbf{a}) = g(x,y,z;\mathbf{a}) = 0, \tag{2}$$

where **a** is the parameters describing the surface with respect to the ORF. For example, the equation for the general quadratic surface is

$$a_{11}x^2 + 2a_{12}xy + 2a_{13}xz + a_{22}y^2 + 2a_{23}yz + a_{33}z^2 + 2a_{14}x + 2a_{24}y + 2a_{34}z + a_{44} = 0. \tag{3}$$

In this case we denote $\mathbf{a} = (a_{11}, a_{12}, \ldots, a_{44})^T$.

II.C Images of an Object Surface Point in Two Image Frames

As shown in Fig. 1.a, P denotes a point on a parameterized 3-D surface of interest. This surface is described by a function in the ORF (see section II.B). The function is uniquely determined by specifying the values of a parameter vector **a**. Point P on the object surface is seen as points having coordinates **s** and **u** in images 1 and 2, respectively. We assume a ***Lambertian reflectance model***. Then the images of point P at **s** and **u** will have the same intensity. The techniques proposed will not apply to specular reflectors, without modification, because the location of points on the object surface at which specular reflection occurs depends on the camera location. Since most surfaces of interest are largely Lambertian, the assumption is a useful one. Hence,

$$I_2(\mathbf{u}) = I_1(\mathbf{s}) \tag{4}$$

where $I_1(\mathbf{u})$ and $I_2(\mathbf{s})$ are the picture functions (image intensity functions) in Frames 1 and 2, respectively. For those cases where the Lambertian assumption does not apply, a possible modified approach is to use an edge map. Here, pixels are given values of 128 or 0 depending on whether they are detected as being edge points or non edge points, respectively. These maps are then smoothed to obtain more continuous arrays, and these are used as though they are regular picture functions in our estimation algorithms. The usefulness of the edge map is that it is a representation of rapid changes in the object surface patterns, and largely unaffected by the presence of some specular component in the object surface. Experiments using edge maps with our algorithm are described in [6].

For simplicity, we use the orthographic projection model [7] for image formation, i.e., all rays from points on the object surface to the camera are roughly parallel. (With slight modification, all of our results can be used with the perspective projection model.) Let $r(1) = (x_1\ y_1\ z_1)^T$ be the coordinates of the 3-D surface point P with respect to CRF1, and $r(2) = (x_2\ y_2\ z_2)^T$ be the coordinates of the point P with respect to CRF2. Then, under the assumption of orthographic projection,

$$\mathbf{s} = (x_1\ y_1)^T, \qquad \mathbf{u} = (x_2\ y_2)^T.$$

If we pick a point **s** in image plane 1, it will correspond to some point P on the 3-D surface. If this point P is also seen in image 2, its image in image plane 2 will occur at some coordinate **u**. Therefore, given some point **s** in image 1, if we want to compute the corresponding image point **u** in image 2 based on the current estimation of **a**, we can:
(i) first, find the 3-D location of the corresponding surface point P;
(ii) then, find the image point **u** corresponding to P.

In step (i), represent the surface point P with respect to CRF1 by $r(1) = (x_1\ y_1\ z_1)^T$. Using equations (1) and (2), the equation of the surface is

$$g(\mathbf{r};\ \mathbf{a}) = g(\ B(1)\,r(1) + r_c(1)\ ;\ \mathbf{a}) = g(\ B(1)(x_1\ y_1\ z_1)^T + r_c(1)\ ;\ \mathbf{a}) = 0. \tag{5}$$

Since the point P resides on the surface, r(1) must satisfy the above equation. Therefore, given $\mathbf{s} = (x_1\ y_1)^T$, we can solve equation (5) for z_1. An example for the spherical surface is given in the next section.

In step (ii), we want to compute **u**. Now that we have obtained r(1) from step (i), using equation (1) we can compute $\mathbf{u} = (x_2\ y_2)^T$ by

$$r(2) = (x_2\ y_2\ z_2)^T = B^T(2)\left[r - r_c(1)\right] = B^T(2)B(1)r(1) + B^T(2)\left[r_c(1) - r_c(2)\right]. \tag{6}$$

Call $C = B^T(2)B(1)$ and $d = B^T(2)\ (r_c(1) - r_c(2))$. Then, partition the C matrix and d vector as:

$$C = \begin{bmatrix} C_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}, \qquad d = \begin{bmatrix} e \\ d_3 \end{bmatrix},$$

where C_{11} is 2×2, c_{12} and c_{21}^T are 2×1, c_{22} is a number, e is 2×1, and d_3 is a number. From the preceding:

$$u = C_{11}s + c_{12}z_1(s,b_1,a) + e. \tag{6a}$$

Combining steps (i) and (ii) above, we denote the functional relationship (6a) between s and u by

$$u = h(s, b, z(s,a)\ ), \tag{7}$$

where the vector b includes b_1 and b_2, and specifies C_{11}, c_{12}, and e.

III.A Estimation of the Parameterized Surface Using Two Images

If we know the camera position, b, and the true surface parameters, a_T, then

$$I_1(s) = I_2(\,h(s,b,z(s,a_T))\,) \tag{8}$$

for each s. Choose a region in image 1. Denote this pixel set in this region by D. Consider the error measure

$$e_D(a) = \sum_{s \in D} \left[I_1(s) - I_2(\,h(s,b,z(s,a))\,)\right]^2. \tag{9}$$

Then $e_D(a)$ is a minimum at $a = a_T$. Our problem is to estimate a_T by minimizing (9) with respect to a.

To estimate a_T that minimizes (9), we choose to use the gradient method as follows:

$$a_{n+1} = a_n - \frac{\partial e_D(a_n)}{\partial a}\Delta_n, \tag{10}$$

where Δ_n depends on $e_D(a_n)$ and $\frac{\partial e_D(a_n)}{\partial a}$ and has magnitude that goes to 0 as n goes to infinity.

There are several ways to compute the gradient $\frac{\partial e_D}{\partial a}$. We present one of the methods used in our experiments. Taking the derivative of (9) with respect to a, we have

$$\frac{\partial e_D}{\partial a} = -2 \sum_{s \in D} \left[I_1(s) - I_2(u)\right]\frac{\partial I_2(u)}{\partial a}, \tag{11}$$

where u is a function of a as shown in (7). Use of the chain rule gives[‡]

$$\frac{\partial I_2(u)}{\partial a} = \frac{\partial I_2(u)}{\partial u}\frac{\partial u}{\partial z}\frac{\partial z(s,\ a)}{\partial a}, \tag{12}$$

where $u = h(s,\ b,\ z(s,a)\ )$ as in equation (7). The first term $\frac{\partial I_2(u)}{\partial u}$ can be computed approximately using the sobel operator. The second term $\frac{\partial u}{\partial z}$ is just a constant provided that we assume the orthographic projection model. This can be shown as follows. From equation (1)

$$r(2) = B^T(2)\left[r - r_c(2)\right], \tag{1}$$

and upon using the notation

$$r(2) = (x_2\ y_2\ z_2)^T, \qquad u = (x_2\ y_2)^T, \qquad r = (x\ y\ z)^T,$$

‡ The notation used here is that $\frac{\partial I_2(u)}{\partial a}$ is a K component row vector, where K is the number of the components in column vector a.

we have $\frac{\partial a}{\partial z} = (B^T(2)_{13}\ \ B^T(2)_{23})^T$, where $B^T(2)_{ij}$ means the ijth element of matrix $B^T(2)$.

In general, it may be inconvenient to express z as an explicit function of a. Hence, we compute the third term by

$$\frac{\partial z(s,\ a)}{\partial a} = \frac{\partial g(x,y,z,a)}{\partial a} \Big/ \frac{\partial g(x,y,z,a)}{\partial z}. \tag{13}$$

III.B An Example: The Sphere

To illustrate the approach, consider a spherical surface described by the equation

$$(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 = R^2. \tag{14}$$

For this surface, z can be solved for explicitly, via

$$z = z_0 \pm (R^2 - (x - x_o)^2 - (y - y_o)^2)^{1/2}. \tag{15}$$

The positive square root is used since the outside surface of the sphere is seen by the camera looking in the negative z direction. Hence,

$$\begin{aligned}(\frac{\partial z(s,\ a)}{\partial a}) &= (\frac{\partial z}{\partial x_o}\ \ \frac{\partial z}{\partial y_o}\ \ \frac{\partial z}{\partial z_o}\ \ \frac{\partial z}{\partial R})\\ &= (\ (x - x_o)/(z - z_o),\ \ (y - y_o)/(z - z_o),\ \ 1,\ \ R/(z - z_o)\ )\end{aligned} \tag{16}$$

and $z - z_o = (R^2 - (x - x_o)^2 - (y - y_o)^2\ )^{1/2}$. The vector $\frac{\partial z}{\partial a}$ can be computed directly from this.

The analogous equations for planes, cylinder and general quadrics are presented in [6].

III.C Algorithm Operation Interpretation

Fig. 1b is useful for illustrating, in two dimensions, the operation of our algorithm for estimating a_t. Spheres in 3-D are shown as circles. Consider the processing of the image patch between points s′ and s″ in Frame 1. This patch is the image of the patch between points p′ and p″ on the true sphere labeled a_t. The same patch on the sphere surface gives rise to the image patch between points u′ and u″ in Frame 2. Now suppose the system's estimation of a_t is $\hat{a}$. The associated sphere is shown. The performance functional for the estimate of a is given by (9) and is computed as follows. The system thinks that the locations on the sphere surface that give rise to the images at points s′ and s″ in Frame 1 are the intersections of the dashed lines, from s′ and s″, with the sphere labeled $\hat{a}$. These sphere surface points would be seen as the images at point $\hat{u}'$ and $\hat{u}''$ in Frame 2. Hence, the system takes the image patch between points $\hat{u}'$ and $\hat{u}''$ in Frame 2 and assumes that the image at each point u in this interval is the same image as the image at a point s in the interval between s′ and s″ in Frame 1. The points u and s are related geometrically as in the figure, or algebraically by (4). Performance functional (9) requires computing this error $I_1(s) - I_2(\ h(s, b, z(s,a))\ )$.

We make the following interesting observations. From the geometry of image formation in Fig. 1b, the varying scale change that maps the image patch over interval [s′, s″] in Frame 1 into the image patch over interval [u′, u″] is seen. Note that both a scale change and a translation are involved in this 2-D illustration.

If the incorrect a is used in computing the performance functional (9), the patch of image used in Frame 2 is that over the interval [$\hat{u}'$, $\hat{u}''$]. Note that this interval is both a shift and a varying scaling of the interval [u′, u″]. If instead of a sphere, we were dealing with a planar surface, the scale change would be constant throughout the image.

IV Estimation of Parametrized Surface Based on a Sequence of Images

Now suppose a sequence of images is available for estimating a_T, the true parameters of the surface. How can best use be made of this data set? In this section we develop a computationally reasonable approximately maximum likelihood estimator (mle) for a_T.

IV.A The Model

The model that we use for the nth image $I_n(u)$, $u \in D_n$, is that of some true picture function $\mu_n(u)$ plus additive white noise having variance σ^2. Hence, $I_n(u)$, $u \in D_n$, is a set of random variables having joint probability density function (pdf)

$$(2\pi\sigma^2)^{-d_n/2} \exp \sum_{u \in D_n} \left\{-(1/2\sigma^2)\left[I_n(u) - \mu_n(u)\right]^2\right\} \tag{17}$$

where d_n is the number of pixels in D_n. We introduce the more compact notation: $u_n(s,a) = h(s,b_n,a)$, and $s_n(u,a) = h^{-1}(u,b_n,a)$, where b_n is the transformation parameters specifying the nth camera position. Let I_n denote the vector of picture function values, i.e., it has components $I_n(u)$, $u \in D_n$. Let μ_n denote the vector having components $\mu_n(u)$, $u \in D_n$. Then (17) is a function of the parameter vector $(\mu_n^T \; \sigma^2 \; a^T)^T$. Because of the Lambertian assumption for image formation,

$$\mu_n(u) = \mu_l(s_n(u,a)). \tag{18}$$

Hence, the μ_n for all n can be specified in terms of μ_l. Then $\alpha = (\mu_l^T \; \sigma^2 \; a^T)^T$ is a parameter vector that specifies the pdf's (17), for all I_n. Since the additive image noise is independent from image to image, the log of the joint pdf of $I_1, \ldots, I_N$ is

$$\begin{aligned} L_N(\alpha) &= \ln p(I_1, I_2, \ldots, I_N \mid \alpha) \\ &= -\left[\sum_{n=1}^{N} d_n/2\right]\left[\ln 2\pi\sigma^2\right] - (1/2\sigma^2)\sum_{n=1}^{N}\left\{\sum_{u \in D_n}\left[I_n(u) - \mu_l(s_n(u,a))\right]^2\right\}. \end{aligned} \tag{19}$$

Our goal is to find $\hat{a}_N$ that maximizes (19). Since this estimate is a maximum likelihood estimate (mle), we know that it has certain desirable properties such as converging to a_T as $\sum_{n=1}^{N} d_n \to \infty$, and having minimum covariance matrix for the error in the estimation of a_T as $\sum_{n=1}^{N} d_n$ becomes large. The difficulty here is that μ_l is a priori unknown. Hence, in order to compute $\hat{a}_N$ we must simultaneously compute $\hat{\mu}_{lN}$, the estimate of μ_l based on Isun1,..., I_N. Though this looks like a formidable computational challenge, it is in fact easily manageable. In [6] we showed that (9), the performance functional we minimize for estimating a_T in the two picture case, is equivalent to (19) for N=2.

IV.B The Asymptotic Representation

As in section III.A, gradient methods can be used for minimizing $-L_N(\alpha)$. A problem here is that N images must be stored and processed simultaneously. This incurs both a great amount of storage and a large amount of processing for each N. An effective approximation for avoiding this storage problem can be had as follows. Let $\underline{I}_N$ denote $I_1, I_2, \ldots, I_N$. In [3] it is shown that

$$p(\underline{I}_N \mid \alpha) \simeq p(\underline{I}_N \mid \hat{\alpha}_N) \exp\left\{-(1/2)(\alpha - \hat{\alpha}_N)^T \Psi(\underline{I}_N, \hat{\alpha}_N)(\alpha - \hat{\alpha}_N)\right\} \tag{20}$$

where the function $\Psi(\underline{I}_N, \hat{\alpha}_n)$ is a K×K matrix having ijth element

$$[\Psi(\underline{I}_N, \hat{\alpha}_N)]_{ij} = -\frac{\partial^2}{\partial\alpha_j \partial\alpha_i} \ln p(\underline{I}_N \mid \alpha)\Big|_{\alpha = \hat{\alpha}_N} \tag{21}$$

and K is the number of components in α. Hence, (20) has a Gaussian shape in α with mean α_N and inverse covariance matrix $\Psi(\underline{I}_N, \hat{\alpha}_N)$.

Now suppose we wish to compute $\hat{\alpha}_{N+1}$. We can write $p(\underline{I}_{N+1} \mid \alpha) = p(\underline{I}_N \mid \alpha) p(I_{N+1} \mid \alpha)$, so that upon using (20), there results

$$L_{N+1}(\alpha) \simeq \left[\sum_{n=1}^{N} \ln p(I_n \mid \hat{\alpha}_N)\right] - \frac{1}{2}(\alpha - \hat{\alpha}_N)^T \Psi(\underline{I}_N, \hat{\alpha}_N)(\alpha - \hat{\alpha}_N) + \ln p(I_{N+1} \mid \alpha) \tag{22}$$

The appeal of (22) is that all the useful information in $\underline{I}_N$ is summarized in the quadric form, i.e., the second term on the right hand side of (22). Notice that only the two rightmost terms in (22) are functions of α. Now $\hat{\alpha}_{N+1}$ can be found approximately as the α that maximizes (22). Gradient descent can be used on (22). The gradient here is simply

$$\frac{-\partial L_{N+1}(\alpha)}{\partial\alpha} \approx \Psi(\underline{I}_N, \hat{\alpha}_N)\cdot(\alpha - \hat{\alpha}_N) - \frac{\partial}{\partial\alpha}\ln p(I_{N+1} \mid \alpha)\,. \tag{23}$$

There is considerable computation here, since there are M^2 components for μ_1 in an M×M pixel patch, and α would therefore have $M^2 + K + 1$ components. A simplification is possible upon realizing that since the dependence of (22) on μ_1 is as a sum of two quadrics in μ_1, a simple explicit value can be found for $\hat{\mu}_{1(N+1)}$ in terms of $\hat{\alpha}_N$, I_{N+1}, $\Psi(\underline{I}_N, \hat{\alpha}_N)$, σ^2, and $\mathbf{a}$. The resulting function to minimize is then a function of only σ^2 and $\mathbf{a}$, hence, only K+1 parameters. Gradient descent can be used for this purpose. This solution is explored in [9]. Though this should provide the most accurate estimate for α_T, for a number of reasons we have minimized a simpler function.

IV.C Approximate Likelihood Maximization

In this section, we treat $I_1(s)$, $s\in D$, as if it were $\mu_1(s)$. Then μ_1 is no longer treated as unknown -- only σ^2 and $\mathbf{a}$ are unknown. If our goal is to estimate $\mathbf{a}$ only, then we do not have to estimate σ^2 since σ^2 gives information only about the accuracy of the estimate for $\mathbf{a}$ (see [6]) but does not affect the value of the estimate for $\mathbf{a}$ (see [9]). Hence, $\alpha = \mathbf{a}$. For practical reasons, instead of letting D_n be an arbitrary subset in image plane n, we proceed analogously to the two image problems in Sec. III.A.. Hence, (17) becomes

$$p(I_n \mid \mathbf{a}) = (2\pi\sigma^2)^{-d_1/2}\exp\left\{\sum_{s\in D_1} -\frac{1}{2\sigma^2}\left[I_n(\mathbf{u}_n(s,\mathbf{a})) - I_1(s)\right]^2\right\}. \tag{24}$$

Then,

$$\begin{aligned} L_{N+1}(\mathbf{a}) \approx \ln p(\underline{I}_N \mid \hat{\mathbf{a}}_N, \sigma^2) - \frac{1}{2}(\mathbf{a} - \hat{\mathbf{a}}_N)^T\Psi(\underline{I}_N, \hat{\mathbf{a}}_N, \sigma^2)(\mathbf{a} - \hat{\mathbf{a}}_N) \\ + \ln p(I_{N+1} \mid \mathbf{a}, \sigma^2)\,. \end{aligned} \tag{25}$$

Now, our goal is to compute $\hat{\mathbf{a}}_{N+1}$, the value of $\mathbf{a}$ that minimizes the negative of (25). We suggest a gradient descent algorithm similar to that used in Sec. III.A. Let $\hat{\mathbf{a}}_{N+1,k}$ denote the estimate for $\mathbf{a}_T$ after the kth iteration in the N+1st stage (i.e., the N+1st stage is that following the input of the N+1st image and prior to the input of the N+2nd image). Then we compute $\hat{\mathbf{a}}_{N+1}$ as the limit of $\hat{\mathbf{a}}_{N+1,k}$ in (26).

$$\hat{\mathbf{a}}_{N+1,k+1} = \hat{\mathbf{a}}_{N+1,k} + \text{SCALE}\cdot\left.\frac{\partial L_{N+1}(\mathbf{a})}{\partial\mathbf{a}}\right|_{\mathbf{a}=\hat{\mathbf{a}}_{N+1,k}}. \tag{26}$$

From (24) and (25),

$$\frac{\partial L_{N+1}(\mathbf{a})}{\partial\mathbf{a}} \approx \Psi(\underline{I}_N,\hat{\mathbf{a}}_N,\sigma^2)(\mathbf{a} - \hat{\mathbf{a}}_N) + \sigma^{-2}\sum_{s\in D_1}\left[I_{N+1}(\mathbf{u}_{N+1}(s,\mathbf{a})) - I_1(s)\right]\frac{\partial I_{N+1}(\mathbf{u}_{N+1}(s,\mathbf{a}))}{\partial\mathbf{a}} \tag{27}$$

Once $\hat{\mathbf{a}}_{N+1}$ is computed, $\Psi(\underline{I}_N, \hat{\mathbf{a}}_N, \sigma^2)$ can be updated to $\Psi(\underline{I}_{N+1}, \hat{\mathbf{a}}_{N+1}, \sigma^2)$ by

$$\Psi(\underline{I}_{N+1}, \hat{\mathbf{a}}_{N+1}, \sigma^2) = \Psi(\underline{I}_N, \hat{\mathbf{a}}_N, \sigma^2) - \left.\frac{\partial^2}{\partial\mathbf{a}\partial\mathbf{a}}\ln p(I_{N+1} \mid \mathbf{a}, \sigma^2)\right|_{\mathbf{a}=\hat{\mathbf{a}}_{N+1}} \tag{28}$$

with $-\frac{\partial^2}{\partial\mathbf{a}\partial\mathbf{a}}\ln p(I_{N+1} \mid \mathbf{a},\sigma^2)$ a matrix having ijth element

$$\sigma^{-2}\sum_{s\in D_1}\left\{\left[I_{N+1}(\mathbf{u}_{N+1}(s,\hat{\mathbf{a}}_{N+1}))-I_1(s)\right]\frac{\partial^2 I_{N+1}(\mathbf{u}_{N+1}(s,\hat{\mathbf{a}}_{N+1}))}{\partial a_j\partial a_i} + \left[\frac{\partial I_{N+1}(\mathbf{u}_{N+1}(s,\hat{\mathbf{a}}_{N+1}))}{\partial a_j}\,\frac{\partial I_{N+1}(\mathbf{u}_{N+1}(s,\hat{\mathbf{a}}_{N+1}))}{\partial a_i}\right]\right\}. \tag{29}$$

For brevity, denote $\Psi(\underline{I}_N, \hat{\mathbf{a}}_N, \sigma^2)$ by Ψ_N. Then the incremental stereo algorithm is summarized as follows:

```
1.  Read image 1.
2.  Set $\Psi_1 = 0$.
3.  For $N \geq 1$.
4.      Read image N+1.
5.      Compute $\hat{a}_N$ by using Eq. 26 iteratively until it converges.
6.      Compute $\Psi_{N+1}$ by Eq. 28.
```

IV.D Experiments With the Algorithm in Section IV.C

Figure 2 shows a sequence of nine computer generated images of a sphere. The images were generated by taking a few images of faces with a solid state T.V. camera, and using the computer to project these images onto a sphere. Using the pattern on the sphere generated in this way, the computer was then used to generate the images that should be seen by a camera at nine locations and with a specified CRF at each location. For this experiment the camera moved along a circular arc of radius 2000 units lying in a horizontal plane. The camera optical axis pointed to the center of this arc, and there was no rotation of the image plane about the optical axis. The angles between the camera optical axes in successive images were 5°. The patch of subimage used in each of the nine images is the region about the left eye of the rightmost face in the image. The patch of subimage is outlined as roughly a small square in white. The parameters specifying the sphere are $(x_0, y_0, z_0)^T$, the sphere center, and R, the sphere radius. In the experiments run, the sphere radius was assumed to be known and only the center was estimated. Table 1 shows the values of $\hat{a}_N$ found. The initial guess used for the sphere center was in error by a little more than the sphere radius of 128 units. The final estimate is in error by roughly two units. The optical axis of the camera moved through an angle of 40° from its first to its last position. These images were noiseless. However, some error is introduced because images are spatially quantized into pixels. Table 2 shows the estimates $\hat{a}_N$ for a more noisy image sequence. Each image here is the image in the corresponding position in Fig. 2 plus white Gaussian noise. The added noise has standard deviation of 5 units (i.e., variance of 25 units). The initial estimate $\hat{a}_1$ used here is also in error by about the sphere radius. The final error, based on nine images, is a little bit more than that in Table 1, but it is small. The accuracy of the algorithm appears to be remarkably good considering the small patches of data used in the estimation. In practice, image 1 would be partitioned into many squares, and a sequence of estimates would be obtained for each. The information obtained from each patch would be optimally combined using the methods presented in [3], thereby greatly improving the accuracy of the estimate of a_T. With the initial error used here, the algorithm in (26) went through about 8-10 iterations to compute $\hat{a}_N$ at each stage.

Figure 3 contains plots of $e_D(a)$, equation (30), as functions of x_0 and y_0, with z_0 held fixed at its true value, -2000.

$$e_D(a) = \sum_{n=1}^{N} \sum_{u \in D_n} \left[I_n(u) - \mu_1(s_n(u,a)) \right]^2, \tag{30}$$

The purpose of these plots is to show how $e_D(a)$, which is the function that must be minimized to maximize (19), narrows in the vicinity of its minimum as the number of images used increases. Since the height of $e_D(a)$, i.e., the distance between its minimum and maximum is an increasing function of the number of images used, we have only plotted the functions in the vicinity of their minima. That is, the plots stop at a height of roughly 3000 units above the minima. The functions shown are based on the use of 2, 6, and 10 images, respectively. It is seen that the functions narrow appreciably in going from the use of two images to the use of 10 images. In Fig. 4, curves of $e_D(a)$ are again shown , but only two images are used in each case. However,, the angle between the pair of camera optical axes varies, with angles of 1°, 5°, and 45° for the three plots shown. Notice how broad and flat the bottom of the curve associated with 1° is, whereas the curve associated with 45° is much narrower, as expected. However, it is still not as narrow as the curve in Fig. 3c where the angle between the optical axes of the first and tenth cameras is 40°. Hence, both the range of angles spanned and the number of images used is important. The other observation of interest is that the functions in Fig. 3 and those in Figs. 4a and 4b are smooth, whereas that in Fig. 4c is not. The multimodal behavior of Fig. 4c is due to the high frequencies in the pattern on the sphere surface. The effect is moderated when the angle between the optical axes of a pair of images is small, and the effect is also suppressed by the averaging that takes place when many more than two images are used.

V Incremental Stereo

A slightly different formulation is to write the joint likelihood for the image differences $I_2 - I_1$, $I_4 - I_3, \ldots, I_N - I_{N-1}$. The joint likelihood can be written as

$$\prod_{n=1}^{N/2} \left[2\pi(2\sigma^2)\right]^{-d_{2n-1}/2} \exp\left\{-\frac{1}{4\sigma^2} \sum_{u \in D_{2n-1}} \left[I_{2n}(u_{2n}(u,a)) - I_{2n-1}(u)\right]^2\right\}. \tag{31}$$

Here, $u_{2n}(u,a)$ in $I_{2n}(u_{2n}(u,a))$ denotes the point in D_{2n} that the point u in D_{2n-1} maps to. The mean value functions $\mu_1(s_n(u,a))$ do not appear here since the expectation of $I_{2n}(u_{2n}(u,a_T)) - I_{2n-1}(u)$ for each u is 0. Also, the variance of this difference for each u is $2\sigma^2$. Then $a^{\dagger}_{N+2}$ is to be chosen to minimize

$$-L^{\dagger}_{N+2}(a) = \sum_{n=1}^{(N+2)/2} \frac{d_{2n-1}}{2} \ln(4\pi\sigma^2) + \sum_{n=1}^{(N+2)/2} \frac{1}{4\sigma^2} \sum_{u \in D_{2n-1}} \left[I_{2n}(u_{2n}(u,a)) - I_{2n-1}(u)\right]^2. \tag{32}$$

Again, it is computationally undesirable to store the N+2 images and also to process all of them simultaneously in order to compute $a^{\dagger}_{N+2}$. Hence, as in Sec. IV.C., we use an asymptotic approximation, Gaussian in a, to represent (31) when computing $a^{\dagger}_{N+2}$.

Table 3 contains the estimates $a^{\dagger}_N$ based on a sequence of images including those in Fig. 2. Note that the angle between the optical axes for the first and last camera positions used for the images in Fig. 2 is 40°. The viewing angle spanned by for the 18 camera positions used in computing Table 3 is 85°. Notice that even with using 18 images---9 pairs of differences---the algorithm in Sec. IV.C is considerably more accurate. The reason is that the algorithm minimizing (32) uses only the differences in pairs of images taken with camera optical axes that are almost parallel. Hence, it is small baseline stereo and suffers many of the disadvantages of the use of optical flow. If the images are noisy, the relative accuracy of this algorithm would probably degrade considerably. It is interesting to note that the size of the angle between the optical axes of the first and last images is not very important here. Rather, improved accuracy comes from using many pairs of images in order to average out the effects of noisy perturbations.

On the other hand, small angle stereo permits computational advantages which we briefly touch upon. If the camera does not move much in going from the (2n-1)th to the (2n)th position, $u_{2n}(u,a)$, where $u \in D_{2n-1}$, is close to u since CRF(2n) is close to CRF(2n-1). Hence, we can use the Taylor series expansion:

$$I_{2n}(u_{2n}(u,a)) \simeq I_{2n}(u) + \left[\frac{\partial I_{2n}(u)}{\partial u}\right]\left[\frac{\partial u_{2n}(u,a)}{\partial b}\right] \Delta b. \tag{33}$$

Thus,

$$[I_{2n}(u_{2n}(u,a)) - I_{2n-1}(u)]^2 \simeq \left\{\left[I_{2n}(u) - I_{2n-1}(u)\right] + \left[\frac{\partial I_{2n}(u)}{\partial u}\right]\left[\frac{\partial u_{2n}(u,a)}{\partial b}\right] \Delta b\right\}^2, \tag{34}$$

where Δb is an incremental vector specifying the incremental rotation and the incremental origin translation for CRF(2n) in term of CRF(2n-1). The desirability of the approximation is that in minimizing (32) with respect to a it is no longer necessary to compute the $u_{2n}(u,a)$ and then the arrays $I_{2n}(v,a)$ and $\left.\frac{\partial I_{2n}(v)}{\partial v}\right|_{v=u_{2n}(u,a)}$ for all $u \in D_{2n-1}$. Rather we can just use the arrays $I_{2n}(u)$ and $\frac{\partial I_{2n}(u)}{\partial u}$, $u \in D_{2n-1}$, directly. This makes for a considerable reduction in required computation. Furthermore, note that when computing the gradient of (34) with respect to a, only the term $\frac{\partial u_{2n}(u,a)}{\partial b}$ is a function of a, and this function is very simple as seen in Eq. 6a.

The final remark of interest is that for the planar surface described in the appendix, the use of Eqs. 6, 34, and A2 (from the appendix) in Eq. 32 permits a simple *explicit* solution for $a^{\dagger}_{N+1}$, the value of a that minimizes (32).

VI Conclusion

In this paper, for the first time the joint likelihood of two or more images as a function of the a priori unknown 3-D surface to be estimated is derived. This permits the full range of Bayesian analysis, estimation, and recognition techniques to be applied to the 3-D surface inferencing problem. In particular, in this paper we develop a recursive

algorithm for the maximum likelihood estimation of a parameterized surface based on a sequence of images taken, perhaps, by a moving camera. This recursive estimator should be significantly more accurate than the use of the extended Kalman filter, since the latter uses a linearization about the Nth stage estimate to compute the N+1st stage estimate whereas we use the complete information in the N+1st image.

APPENDIX: The Plane

We derive the expression for the vector $\partial z/\partial a$ for a plane. Note that there are a number of different sets of parameters that can be used for representing a plane (or a cylinder, or a more general surface). We use the canonical parameterization in this section. We use the equation

$$0 = g(x,y,z) = \beta_1 x + \beta_2 y + \beta_3 z - d \tag{A1}$$

subject to the constraint

$$0 = f(x,y,z) = \beta_1^2 + \beta_2^2 + \beta_3^2 - 1 \tag{A1a}$$

Note, $|d|$ is the distance from the plane to the origin in this representation. It is assumed that the plane is in general position, because if, e.g., $\beta_3 = 0$, then the plane normal is orthogonal to the first camera's optical axis, and the plane surface is not seen by the first camera since the camera then sees only the plane's edge. Eq. (A1a) can be used to solve for β_3 in terms of β_1 and β_2. Hence we can take a to be $a^T = (\beta_1,\beta_2,d)$. Now $\frac{\partial z}{\partial a} = -\frac{\partial g/\partial a}{\partial g/\partial z}$. Using (A1a), we get $\frac{\partial \beta_3}{\partial \beta_1} = -\frac{\partial f/\partial \beta_1}{\partial f/\partial \beta_3} = \frac{-2\beta_1}{2\beta_3} = -\frac{\beta_1}{\beta_3}$. Similarly, $\frac{\partial \beta_3}{\partial \beta_2} = -\frac{\partial f/\partial \beta_2}{\partial f/\partial \beta_3} = -\frac{\beta_2}{\beta_3}$. Hence, $\frac{\partial g}{\partial z} = \beta_3$

$$\frac{\partial g}{\partial a} = \begin{cases} \frac{\partial g}{\partial \beta_1} = x + z\frac{\partial \beta_3}{\partial \beta_1} = x - z\frac{\beta_1}{\beta_3} = \frac{\beta_3 x - \beta_1 z}{\beta_3} \\ \frac{\partial g}{\partial \beta_2} = y + z\frac{\partial \beta_3}{\partial \beta_2} = \frac{\beta_3 y - \beta_2 z}{\beta_3} \\ \frac{\partial g}{\partial d} = -1 \end{cases}$$

Thus,

$$\frac{\partial z}{\partial a} = \left[\frac{\beta_1 z - \beta_3 x}{\beta_3^2}, \frac{\beta_2 z - \beta_3 y}{\beta_3^2}, \frac{1}{\beta_3}\right]. \tag{A2}$$

ACKNOWLEDGEMENT

Partial support was provided by NSF Grant #ECS-81-19676, ARO Grant #DAA-LO3-86-K-0011, the IBM Co., and by the University of Buenos Aires.

REFERENCES

[1] D.H. Ballard, C.M. Brown, Computer Vision, Prentice Hall, Englewood Cliffs, N. J., 1982.

[2] P.J. Besl, R.C. Jain, "Invariant Surface Characteristics for 3D Object Reconstruction in Range Images," Computer Vision, Graphics, and Image Processing, 33, 1986, pp 33-80.

[3] R.M. Bolle, D.B. Cooper, "On Optimally Combining Pieces of Information, with Application to Estimating 3-D Complex-Object Position from Range Data," IEEE Trans. on PAMI, Sept. 1986, pp 619-638.

[4] B. Cernuschi-Frias, P.N. Belhumeur, D.B. Cooper, "Estimating and Recognizing Parameterized 3-D Objects Using a Moving Camera," Proc. IEEE Comp. Soc. Conf. On Computer Vision And Pattern Recognition, San Francisco, June 1985, pp 167-171.

[5] B. Cernuschi-Frias, D.B. Cooper, F.G. Amblard, "Estimation by Multiple Views of Outdoor Terrain Modeled by Stochastic Processes," to appear in Proc. SPIE Cambridge Symposium on Optical and Optoelectronic Engineering: Intelligent Robots and Computer Vision, Cambridge, Mass., Oct. 28-31, 1986.

[6] B. Cernuschi-Frias, D.B. Cooper, P.N. Belhumeur, Y.P. Hung, "Toward A Bayesian Theory for Estimating and Recognizing Parameterized 3-D Objects Using Two or More Images Taken From Different Positions," under review for journal publication. Also, Brown University, Division Of Engineering, LEMS Tech Report #32, December, 1986.

[7] D. Duda, P. Hart, Pattern Classification And Scene Analysis, John Wiley, 1973, p. 381.

[8] O.D. Faugeras, N. Ayache, B. Faverjon, "Building Visual Maps by Combining Noisy Stereo Measurements," Prc. 1986 IEEE Conf. On Robotics And Automation, San Francisco, April 7-10, 1986, pp 1433-1438.

[9] Y.P. Hung, "Maximum Likelihood Estimation of Parameterized Surfaces In 3-D space Using A Sequence of Images," M.Sc. thesis, 1987, Brown University, Division of Engineering.

[10] Y. Ohta, T. Kanade, "Stereo by Intra-and Inter-Scanline Search Using Dynamic Programming," IEEE Trans. on PAMI, March 1985, pp 139-154.

[11] A.M. Waxman, K. Wohn, "Image Flow Theory: A Framework For 3-D Inference From Time-Varying Imagery," a chapter to appear in Advances In Computer Vision, ed. C. Brown (Erlbaum Publishers).

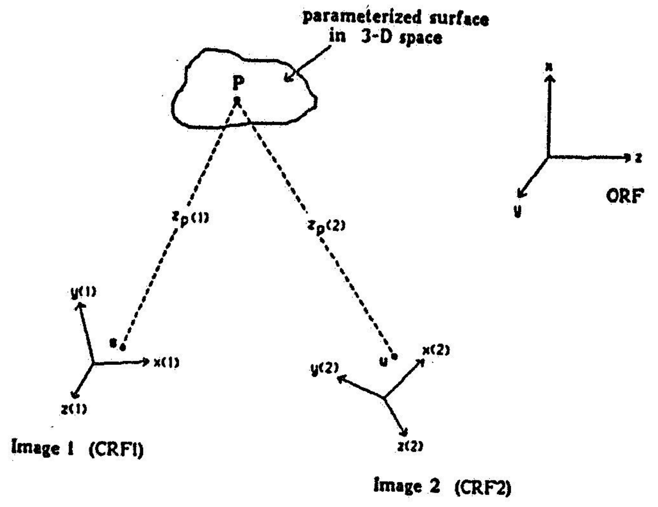


Figure 1a

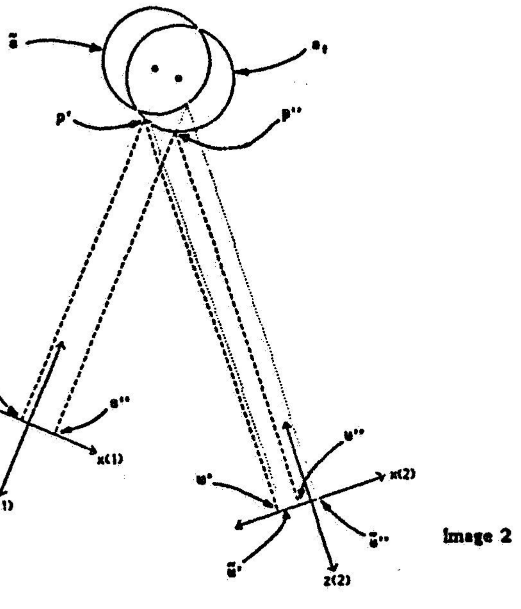


Figure 1b

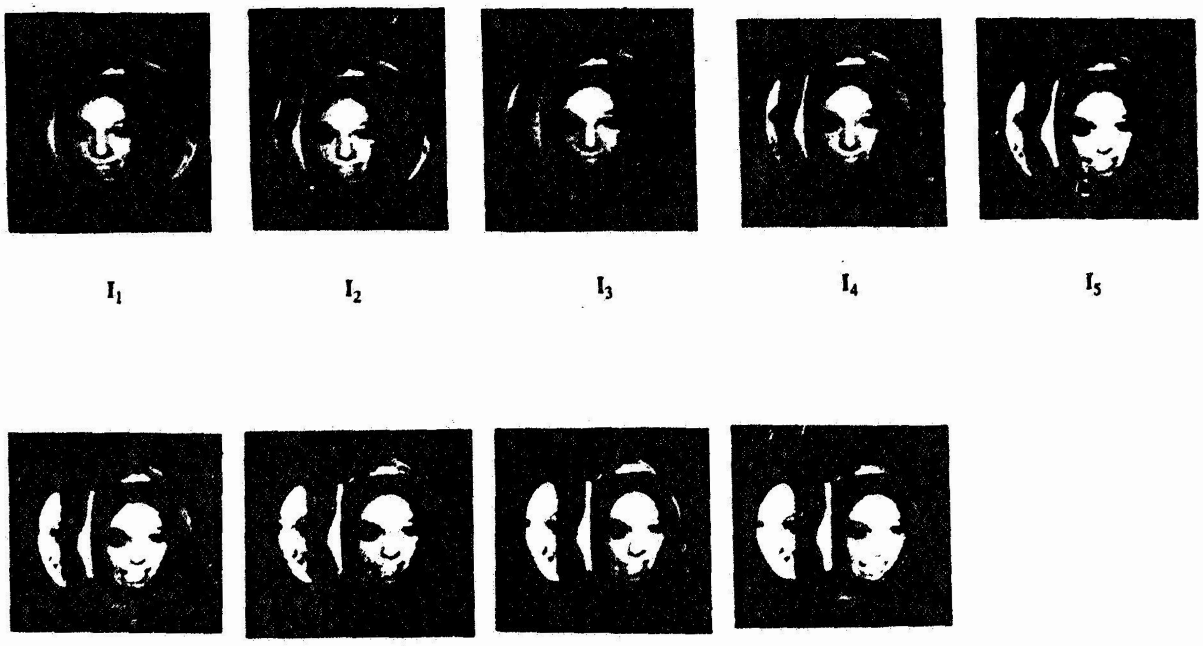


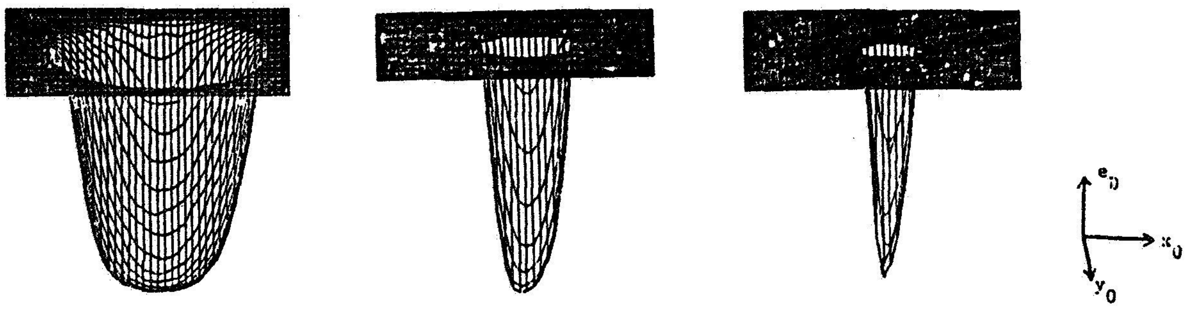
Figure 2

# of image	$\hat{\mathbf{a}}_N$			
	x_0	y_0	z_0	R
initial $\hat{\mathbf{a}}_1$	60.0	60.0	-2120.0	128
2	16.9	9.7	-2001.4	128
3	11.4	6.9	-2002.3	128
4	7.4	4.1	-2001.2	128
5	4.3	2.9	-2000.8	128
6	3.1	2.1	-2000.5	128
7	1.7	1.9	-2000.1	128
8	1.8	1.8	-2000.2	128
9	-0.6	1.9	-2000.1	128
true **a**	0.0	0.0	-2000.0	128

Table 1

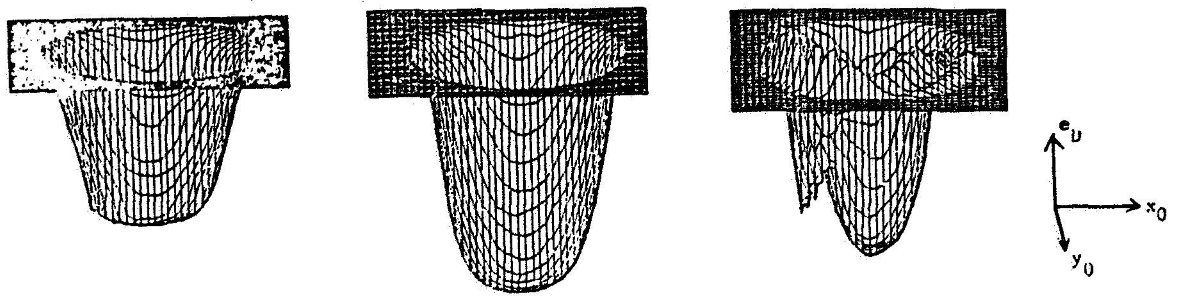
# of image	$\hat{\mathbf{a}}_N$			
	x_0	y_0	z_0	R
initial $\hat{\mathbf{a}}_1$	50.0	50.0	-2080.0	128
2	11.0	6.0	-1993.0	128
3	2.7	-1.6	-1993.6	128
4	-0.6	-13.4	-2000.1	128
5	0.7	-1.7	-2001.9	128
6	0.8	8.3	-2002.4	128
7	1.1	1.4	-2001.8	128
8	0.6	1.6	-2001.5	128
9	1.0	1.5	-2001.3	128
true **a**	0.0	0.0	-2000.0	128

Table 2



(a) based on I_1, I_2 (b) based on $I_1, I_2, \ldots, I_6$ (c) based on $I_1, I_2, \ldots, I_{10}$.

Fig. 3 Error function



(a) with 1° between optical axes (b) with 5° between optical axes (c) with 45° between optical axes

Fig. 4 Error function based on two images

number of images used	$\hat{a}$			
	x_0	y_0	z_0	R
initial $\hat{a}$	50.0	50.0	-2080.0	128
2	26.8	20.3	-2001.8	128
4	16.9	13.3	-2001.1	128
6	10.1	10.4	-2001.4	128
8	8.6	8.8	-2001.2	128
10	10.3	8.9	-1999.9	128
12	5.3	4.9	-2002.1	128
14	5.5	1.3	-2000.4	128
16	5.9	-1.4	-2001.0	128
18	-0.8	-2.2	-2001.9	128
true **a**	0.0	0.0	-2000.0	128

Table 3

511-60
98
188131

The Architecture of a Video Image Processor for the Space Station

S. Yalamanchili, D. Lee, K. Fritze, T. Carpenter, and K. Hoyme
Honeywell Systems and Research Center
Minneapolis, MN 55418

N. Murray
NASA Langley Research Center
Hampton, VA 23665-5225

HY. 989092

N92-10891

is described.

Abstract

~~This paper describes~~ the architecture of a video image processor for space station applications. The architecture was derived from a study of the requirements of algorithms that are necessary to produce the desired functionality of many of these applications. Architectural options were selected based on a simulation of the execution of these algorithms on various architectural organizations. A great deal of emphasis was placed on the ability of the system to evolve and grow over the lifetime of the space station. The result is a hierarchical parallel architecture that is characterized by high level language programmability, modularity, extensibility and can meet the required performance goals.

1 Introduction

A major goal in the design and deployment of the NASA space station is to enable crew members to effectively and efficiently use the resources of the space station. The number of anticipated scientific and commercial missions will place a heavy demand on these resources, one of which is crew time. Thus, facilities that enable crew members to perform their tasks efficiently, effectively, and safely are critical to the success of the space station. This paper describes the utility and feasibility of providing crew members with one such facility - a video image processor (VIP).

Initially, a crew member will directly control and be interactively involved with most activities, such as inspection, docking, experiment monitoring, and control. One of the problems with the man-in-the-loop scenario is that the human operator is frequently performing repetitive observations and control functions that do not exploit the unique capabilities of a human in space, namely, decision making, supervision, and creative thinking. Repetitive tasks are ideally suited for automation by machine. Such automation would free crew members for more demanding tasks as well make more efficient use of their time. An important technology central to automation and robotics is image processing. Video images may be processed to improve the quality for viewing purposes, provide cues in assisting an operator in some task, or provide some information to control other devices or alert the crew when necessary, as in automatic experiment monitoring.

Early on in the study [1], a surprisingly large number of applications were found that could benefit from the availability of an on-board VIP. The required algorithms and architectures necessary to support a VIP were found to be mature enough to make the concept of a VIP feasible. An examination of the functional requirements of image processing algorithms and the capabilities of current and future processors resulted in the conceptual design of a hierarchically structured, parallel architecture for a VIP. This paper reports the results of an effort to refine this conceptual view via simulation. The simulation studies were based on the requirements derived from an analysis of algorithms for space station applications. The design and validation of these algorithms are discussed in a companion paper [2].

2 Role of a VIP in the Space Station

The principal goal of a VIP is to increase the efficiency with which the space station resources are used. This would be achieved by automating certain tasks with the VIP as well as making some tasks more efficient. Although detailed requirements for various systems on the space station are still being developed, it is evident that a number of existing requirements support the concept of an on-board VIP. One requirement is that missions ~~should be~~ performed in a timely and safe manner. Missions include user experiments, user production activities, satellite servicing, and housekeeping tasks. A VIP may enable a crew member to perform more activities from a central location, such as a workstation. This would mean fewer extra vehicular activities (EVAs), which in turn makes the crew member more efficient and, in many instances, considerably safer. In addition, there are tasks that can be performed by a VIP that would result in faster execution (for example, providing automatic camera control), make the task easier (filtering of transmitted video containing noise), or eliminate the active participation of a crew member altogether (automatic experiment monitoring).

Another requirement relates to space station autonomy. Autonomy may be interpreted in two ways. The first interpretation is that an activity that can proceed autonomously from human interaction. A VIP helps in this case by performing functions relating to machine vision, freeing the user from constant interaction. The second is that the space station should operate as autonomous from ground support as feasible. In this instance, a VIP may be used in several ways. It can compress image data, allowing the relatively limited on-board storage capability to be used more efficiently. Thus, requests for data are more likely to be satisfied at the station without accessing ground-based archives. In this manner, a VIP can increase crew efficiency, making it less likely that ground-based personnel would be required to support the workload.

A related requirement is that the crew time necessary for housekeeping tasks should be minimized. As shown in the space station mission requirements report, one of the constraints on the number of active payloads will be the number of crew hours available to perform payload-related tasks. The assumption was made that with a crew of six, the equivalent of one person would be required just to perform housekeeping

PAGE 84 INTENTIONALLY BLANK

chores. A VIP could help reduce this time by automating or supporting housekeeping activities, such as rendezvous and docking, space station inspection, and maintenance. During our examination of potential applications of a VIP [1], we generated the following set of tasks that could potentially utilize a VIP for increased safety, autonomy and efficiency.

- Construction
- Satellite servicing
- Rendezvous and Proximity Operations
- Docking
- Inspection
- Maintenance and Repair
- Payload Delivery and Retrieval
- Experiment Monitoring
- Data Management and Communications
- Training

A more detailed discussion of what role a VIP might play in each of these tasks can be found elswhere [1]. Finally a VIP is impacted by a need to evolve with the space station, primarily since it will not be possible to plan and accommodate all future processing needs. It also has to be compatible with the size, weight and power budgets that are constrained by the capabilities of the power generation subsystem and the payload capacity of the space transportation system.

3 VIP Algorithms

There were two goals for the algorithm selection process. First, the image processing techniques required must be mature and reliable, enabling a high degree of confidence in obtaining the desired functionality. This is particularly important due to the unique set of environmental, lighting and imaging constraints under which space imagery is acquired. Secondly, the algorithm suite should benefit a large number of applications. A cross reference between six major classes of image processing algorithms and the eight generic classes of space station applications is shown in figure 1.

These six families do not represent the entire breadth of the state of the art in image processing, but most of the image processing algorithms required for the automation of space station tasks belong to one of these families. In addition, algorithms in each of these categories are sufficiently mature for the design and build of a prototype system. This prototype system could be semiautonomous in that it could perform the majority of the data reduction necessary for a specific task and an operator would be required for verification/confirmation of the actions of the VIP. Typical applications in which a VIP may perform a task in semiautonomous mode are image enhancement/filtering, intelligent bandwidth reduction, and object velocity estimation for proximity operations. A more detailed discussion is presented in a companion paper [2].

4 VIP Architecture

As a result of the VIP I study, we recommended an architecture for the VIP, shown in Figure 2. It consists of a multiple-SIMD organization (level 1) followed by a multiprocessor organization (level 2). The multiprocessor will be designed to allow for the addition of processors specifically suited for symbolic processing, e.g., rule-based inference processing. The level-1 system may be viewed as a sequence of array processors. The first processor receives video data from the network interface. Each array processor stage may implement a specific image function, such as detector compensation, gray scale stretch, or digital filtering. The processed image may then be transferred to an image memory, which forms the input to another array processor implementing another image function, or may be transferred to the image memory of processors that perform the next higher level of processing (level 2 and level 3). They consist of more flexible multiprocessor systems that can be used to compute descriptions of areas of the image, e.g., regions of interest, boundary codes, statistics, etc. Processing at the higher levels may primarily deal with arrays of real or integer data (e.g., tracking and position estimation) or symbolic data (e.g., relational descriptions). These two types of processing are fundamentally different, but both are required at this level of processing. Our approach is to efficiently accommodate both modes of processing, loosely coupled through the use of a partitioned global address space. Architectures specifically suited for artificial intelligence execution are not considered at this time because of the immaturity of the concept. However, as research continues in this area and in artificial intelligence algorithms for image understanding, the addition of such processors may be allowed during the growth phase of the VIP.

The specific choice of a building block for each level must result in an overall organization that satisfies the constraints identified in the VIP I study. One set of constraints is due to the requirements of the Initial Operating Configuration (IOC). Depending upon the technology freeze date for IOC, current hardware, software, system and algorithm technology may not allow the development of a fully functional VIP within the anticipated size, weight and power constraints. The issue therefore is in phasing: being able to use that part of VIP that is useful and currently feasible, while provisions are made to allow it to evolve into a fully functional VIP. Certain critical portions (such as the level-1 architecture) may be included at IOC to perform those functions that are useful for the man-in-the-loop scenario. Then, during the growth of the space station, additional functionality could be provided using more advanced and stable technology, algorithms, and perhaps architectures. Phased [illegible]lementation requires features of programmability, modularity, and field expansibility. The latter includes provisions for integrating special-purpose devices into the architecture as their need becomes evident and their implementation becomes viable. Further, the implementation technology and architecture must be sufficiently mature to be considered for deployment. The VIP architecture proposed in this program is amenable to all of these constraints.

4.1 Level 1 Architecture

The VIP I study called for a synchronous parallel architecture that operated in single instruction multiple data stream (SIMD) mode and delivered in excess of 600 million operations per second (MOPS) performance. Furthermore, each processor would have access to neighboring processors' memories and would be microcoded. The rationale for these constraints can be found in the VIP I final report [1].

Our choice of the basic building block for the level-1 architecture is the Electro-Optical Signal Processor (EOSP) developed by Honeywell [3]. Its organization satisfies all of the above mentioned constraints. In addition, it possesses a number of other important features that make it a good choice for a VIP. The EOSP architecture was derived in a top-down manner from the requirements of real-time image processing algorithms. The result is a very high speed integrated circuit (VHSIC) chip set that can deliver up to 25 MOPS per processing element (PE) for low-level image processing tasks. Thirty-two PEs constitute a single stage of the EOSP, resulting in a computation rate of 800 MOPS per stage. It is optimized for image processing functions that are characterized by large volumes of data and repetitive arithmetic and logical operations over small neighborhoods of an image. Unlike many early image processing architectures, issues concerning the interface to different sensors and the implementation of image input/output (I/O) were also addressed early in the development. Thus, the EOSP is optimized to provide high throughput for raster scan imaging devices. Any algorithm that exhibits concurrency at the pixel level can be efficiently implemented on the EOSP.

The organization of the EOSP is illustrated in Figure 3. The architecture consists of a linear array of identical PEs, each with its own memory, controlled by a single common controller. This SIMD architecture minimizes the control overhead per PE, thus achieving extremely high computational rates within a very compact processor. In its current form, each PE has a 128-byte input buffer and a 128-byte output buffer. Local memory consists of 512 bytes accessed by a 16-bit arithmetic logic unit (ALU). Each of the I/O buffers is externally clocked. Thus, it is possible for data transfer into the input buffer, data transfer out of the output buffer, and processing of local memory contents all to be occurring simultaneously. This allows for a pipelined mode of operation in which images may be processed in real-time with storage requirements independent of image size. The EOSP architecture operates on an image on a line-by-line basis. Each image line is evenly distributed among the input buffers of the PEs and transferred to local memory. A sufficient number of consecutive image lines is stored to enable one line of image output to be computed. In the configuration of the above example, one line of a K x K window function can be computed by all the processors in parallel. Each processor computes M pixels of the output line. Next a new input line can be acquired, and one line of the computed image can be output. In this manner, one line worth of results is computed and output for every input image line from the sensor. This has the effect of "sliding" a K x K window over the input image. Data from the input buffer are transferred to the individual PE memories in parallel at the end of the scan line. Processed results in the PE memory, computed during the previous line input, are transferred simultaneously to the corresponding output buffers. Buffered results are read out synchronously with the input data entering during the next scan line. Input and output can be double-buffered for sensors that possess no dead time between lines (e.g., retrace time). This provides a great deal of flexibility in interfacing the EOSP to different types of sensors and architectures. Such a feature is especially attractive for the VIP since the functionality of the video network interface (VNI) (input to level-1 architecture) and the details of the level-2 architecture (output from the level-1 architecture) are subject to change. This feature is even more important if the VIP is to be deployed as the level-1 architecture only and is to subsequently evolve to include the level-2 architecture later in the life of the space station.

Sizing an EOSP system is done with respect to three features - processing throughput requirements, memory requirements and the I/O requirements. Whichever feature is the most demanding in terms of the number of processors required, dictates the size of the EOSP system. This essentially accounts for the fact that some applications may be throughput bound versus I/O bound or memory bound. Several examples are illustrated in Table 1. All pixel and neighborhood operations will be implemented in the level 1 architecture. This includes the color image enhancement algorithms [2]. Functioning brassboard versions of the EOSP PEs are available today. The technology and architecture can be considered to be mature by any future technology freeze date. Moreover, a great deal of familiarity with the EOSP systems has been obtained, establishing a degree of confidence in the ability to meet the projected performance goals. Experience has been gained and lessons learned in the design of the EOSP. For this and the reasons cited above, the EOSP architecture is an excellent choice as the building block for the VIP level-1 architecture.

4.2 Level 2 Architecture

Our earlier studies indicated that this level would require an 8 - 16 processor system delivering about 100- 200 MOPS with distributed task allocation, scheduling and synchronization. To understand the characteristics of the level-2 architecture, one needs to understand the algorithms that will be executed. The granularity of parallelism is relatively large (compared to those executed in the level-1 architecture), resulting in a number of concurrently executing tasks. The processing within a task is highly data dependent. As a result, interactions between tasks should be asynchronous. The volume of intertask communication is highly variable and can become the principal determinant of performance [3-4]. Thus, the first issue is the choice of interconnection topology. Once this has been chosen based on the requirements of the VIP algorithms, the architecture may be examined in greater detail to address issues of protocols, processor-specific features, and operating system features. The choices are limited only by one's imagination. However, we chose topologies that, in some sense, occur at extreme points in the spectrum of performance that interconnection networks can provide. At the same time, the choices were filtered by factors such as maturity, available experience with them, and how well we understood them. Our choice of families of topologies to investigate were multiple buses, hypercubes, and braided rings. These topologies are illustrated in figure 4

The next issue is one of analysis techniques. The level-1 architecture exploited fine grain parallelism in a synchronous mode of operation. Further, the algorithms are largely data independent. With such a fine understanding of the implementation of the computations, it is possible to analytically evaluate the architectural options. That is not the case with the level-2 architecture and algorithms. The high degree of variability in the processing and communication requirements indicate that simulation is an appropriate means to determine the proper topologies. The Architecture Design and Assessment System (ADAS) tool set developed at Research Triangle Institute was used for this purpose. The tool set includes facilities for constructing models of communicating parallel tasks and parallel architectures. Further, tools are available to map communicating sequential tasks onto specific architectures and evaluate the performance of such a hardware/software system.

4.2.1 Simulation

The objectives for performing the simulation are multifold. First, we would like to verify that the proposed architecture design can meet the system throughput requirements, and that the specified image processing algorithms can be executed within the given time frame. Second, we want to compare the performance of several proposed architectures and topologies and analyze how they perform in executing the different algorithms. This would then provide guidance in selecting the appropriate architecture approach for the VIP design. Finally, we would like to use simulation as a tool to refine the architecture design. By varying the system size and characteristics, one can perform tradeoffs not only between interconnect topologies, but also in the number of processors and buses, processor speeds, and bus bandwidths. The ultimate objective is to enable us to select and derive a suitable architecture for the VIP design. The parameters we have chosen to study for the VIP simulation effort are categorized as follows.

- Network topology - Six interconnect network topologies were simulated: multiple buses with one two and three buses, hypercube, unidirectional and bidirectional braided rings.
- Communication bandwidth - Three separate bus speeds were used in the simulation: 2, 5, and 10 Mbytes/sec.
- Processor throughput - Three separate processor throughputs were used in the simulation: 2, 5, and 10 million instructions per second (MIPS).
- System size - System sizes of 4, 8 and 16 processors were considered.

In the description of the simulation, buses will be used to refer to both the multiple-access shared media, such as time shared buses, as well as point-to-point links, such as those used in the hypercube and ring organizations. The level 2 implements the components of the tracking and bandwidth reduction algorithm. Each of the computationally intensive components of this algorithm was studied in greater detail and parallel versions of these algorithms were derived and modelled with ADAS. These were,

- Monochrome Segmentation
- Boundary Tracing
- Linearity Filter
- Connected Components
- Silhouette Matching

For each of the above algorithms, conservative requirements on image resolution and other algorithm-specific parameters (e.g., size and number of objects) have been assumed in constructing the software graphs. Both of the above software and hardware systems are modelled in ADAS with directed graphs consisting of nodes interconnected by directed arcs. Nodes represent individual software operations or hardware functional elements, while arcs represent data flow between software operations or hardware components. The presence or absence of data or control is represented by tokens on the arcs. When an input condition is satisfied by the presence of specific patterns of tokens on the input arc a node "fires". It fires for some period of time after which tokens may be placed on some output arcs probably enabling another node. Once software and hardware graphs have been developed, the software graph is mapped onto the hardware graph to produce a constrained software graph. Since the software graph represents the algorithm executed by the hardware, the order in which the software graph nodes fire is determined by the structure of the underlying hardware graph. In particular, software nodes mapped onto the same hardware nodes can only be executed one at a time. Nodes represent the execution of a computation (transfer of data). The firing delays are therefore functions of the volume of computation (data) and the processor speed (link or bus bandwidth). The simulation sequence considers the range of values for processor speeds (link or bus bandwidths). Some examples of hardware and software graphs are shown in figure 5. The simulation sequence proceeds as follows.

1. Construct software and hardware graphs. The software graphs represent the image processing algorithms to be executed, while the hardware graphs represent the architectures and constraints of the hardware system.
2. Place appropriate weights on software nodes. These weights include the various assumed characteristics, such as delays, amount of processing requirements, processor throughputs, and network link bandwidths.
3. Constrain the software graph execution by mapping the software graph to the hardware graph. This involves assigning various software modules (algorithms) to the different hardware modules (nodes).
4. Execute the constrained software graph and collect execution statistics.
5. Modify the weights in step 2 to effect a change in the parameters of interest and repeat the sequence.

For the purpose of evaluating the results, the following performance measures were generated by the simulation.

- Latency - This is the time for one execution of the complete software graph (algorithm).
- Average processor utilization - This is the average percent of execution time [illegible] processors are busy.

- **Maximum processor utilization - This measure is the maximum percent of execution time that a particular processor is busy. It identifies the presence of bottlenecks.**
- **Variance of processor utilization - This provides a measure of balance in processor utilization and thus, the distribution of the computation load.**
- **Average bus utilization - This is the average percent of execution time the buses are being used.**
- **Maximum bus utilization - This identifies the presence of communication bottlenecks.**
- **Variance of bus utilization - This measure indicates the distribution of the communication load.**

To control the ADAS simulation sequence and facilitate the generation of the performance measure statistics, a simulation manager was developed. The simulation manager is the core of the simulation management facility. It essentially controls the iterative execution of the simulation. The details of the simulation management facility can be found in [6].

4.2.2 Simulation Analysis

To facilitate the analysis of the simulation data in selecting a suitable VIP architecture, we decided to evaluate the performance of the various designs based on the following performance metrics.

- Low latency - Latency is the major criteria in evaluating the performance of a design. The system throughput must be above some minimum threshold in order to satisfy the basic timing and processing requirements. Beyond that threshold, low latency may be traded off against other considerations.
- Balanced processor utilization - The preference here is to evenly distribute the processing load among the processors as much as possible, thereby avoiding the presence of bottlenecks and reducing the severity of single-point failures. This can also serve as an indication of how growth and fault tolerance can easily be achieved with the design.
- Balanced bus utilization - The preference here is to avoid communication bottlenecks and severity of single-point failures. Again, this can serve as an indication of the ease with which fault tolerance and future growth may be accommodated.
- Latency and utilization improvement - This is the differential of the latency or utilization as a function of some architectural parameter. This measure is used to identify points of diminishing returns. For example, a doubling of processor speed may produce only a 2% decrease in latency. In that case, the cost of designing a faster processor may not be worth the added speedup. A similar argument can be made for utilization and, in fact, for most parameters. Another view is that this measure indicates the sensitivity of the latency and utilization metrics to various architectural parameters.

In addition to the above performance metrics, we also made the following empirical assumptions concerning the VIP design requirements.

- The design shall provide a processing throughput margin of 100%.
- The design shall provide a communication bandwidth margin of 100%. These first two assumptions allow for growth in algorithmic requirements and other unexpected overheads.
- The design shall allow the presence of spare processors and spare buses. This enables the design to provide for fault tolerance as well as growth capabilities.
- The VIP design shall execute the tracking and bandwidth reduction algorithm at the rate of about one image per second. This assumption is more of a desire than a requirement. In reality, considering the anticipated applications of VIP in the space station, a processing rate of one image per every few seconds may even be acceptable for most applications.

With the above initial assumptions and performance metrics in mind, the simulation data were analyzed and evaluated. A software graph for the tracking and bandwidth reduction algorithm was constructed, and its execution was simulated on the various architectural organizations. This includes parallelized versions of the selected components. The size of the search space of the architectural alternatives is fairly large. There are six organizations - three for the bus-based systems, one for the hypercubes, one for the unidirectional rings, and one for the bidirectional rings. For each organization, there are three system sizes (4, 8, and 16 PEs), three processor speeds (2, 5, and 10 MIPS), and three bus bandwidths (2, 5, and 10 Mbytes/sec). Thus, there are (6x3x3x3) or 162 distinct possible architectural solutions in this formulation. For each possible architectural solution, the parameters of interest are measured and tabulated. These results were examined manually to apply the chosen metrics and select acceptable solutions. The result reveals that a configuration with 16 processors, each with a processing speed of 10 MIPS and a dual-bus network with bus speeds of 5 Mbytes/sec, comes closest to meeting all of the empirical assumptions and performance metrics mentioned above. The simulation performance data for this architecture are summarized in Table 2.

The simulation data also indicate that the hypercube configuration ($N = 4$), with 16 processors at 10 MIPS each and bus speeds of 5 Mbytes/sec, is also a viable alternative. The final latency value for configurations with the hypercube design is shown in Table 3. Currently, the bus-based approach is preferable to the hypercube approach mainly because it is a relatively more mature and well-understood architecture. In this respect, the bus-based approach represents a low-risk approach. While the hypercube technology has now become a commercially viable product, improvements are rapid and continuous. The network is inherently fault tolerant through the presence of mutiple paths between nodes, but it is not immediately obvious how that feature may be efficiently exploited. The area that needs the most attention is operating system support. Efficient internode communication and global resource allocation strategies are lacking and are the focus of several research efforts by both commercial and academic organizations. By comparison, software in general, and operating systems in particular, are much more mature in bus-based systems. Further, increases in performance by the addition of one, two, or a small number of modules are straightforward in bus-based systems. Generally, the number of modules is doubled to maintain the connectivity of the hypercube. The addition of a smaller number of processors is not straightforward. Thus, while simulation experiments indicate that the hypercube is an acceptable solution, practical considerations indicate that bus-based simulations are preferable. Hence, the 16-processor, two-bus system is the choice for a VIP.

4.3 System Issues

System issues can now be addressed in more detail with respect to this specific organization. System issues relate to three aspects of the VIP. The first is the interaction of the VIP with its environment. This is defined by the functionality of the VNI. The second concerns the software requirements, and the third, the hardware requirements.

4.3.1 Interaction with Environment

The VIP is intended to support bidirectional transfer of video data to and from devices on the space station. The VIP processes raw video data from a variety of video sources - including video cameras, video storage devices, and uplink video - and transfers processed, filtered, and enhanced images to various sinking devices on the space station. In order to specify the functionality of the VNI, it is necessary to make some assumptions about the operating environment. For example, what is the nature and frequency of traffic to and from the VIP? It is clearly infeasible to consider all possibilities. Therefore, we focus on what we feel will be the most prevalent scenario for the use of a VIP: a crew member controlling and using the VIP from a multipurpose applications console (MPAC). For example, cameras possibly mounted outside the space station may transmit images to the MPAC. These images may be redirected from the MPAC to the VIP for enhancement for viewing purposes. Alternatively, the VIP could receive images directly from cameras (under MPAC control) and relay results to the MPAC on detection of a specific event, e.g., in automatic experiment monitoring. In such a scenario, the functionality of the VNI would be determined by the nature of the interaction with the MPAC and by the operation and type of communications media between the MPAC and the VIP.

The MPAC will be one of the primary interactive display devices on the space station. Images will be displayed in the video/graphic/text application display area of the MPAC, and the console will present a mixture of information types, such as graphic, tabular, textual, video, discrete, etc. The advanced Work Package 2 implementation guidelines [7] demonstrate a preference for the display of color image data. However, the capability must exist for handling both color and monochrome video data formats. From the point of view of the interaction with a VIP, it is assumed that the MPAC will provide for the buffering of processed images since most functions are not processed at the image data rate of the MPAC, and graphics and image database functions will not be provided by the VIP.

The space station data management system (DMS) can support the requirements for bidirectional communication between the VIP and MPACs. Data transfers between the VIP and the MPAC involve the transfer of commands and images from the MPAC to the VIP and status from the VIP to the MPAC. Commands take the form of enable/disable for the VIP, diagnostic commands, as well as a selection of algorithms. The volume of communications for such a transfer is expected to be low, 200 to 300 bytes every 1/15th second. Thus, tolerable network latencies are determined by the interactive nature of the processing. It is the image transfers that place demands on the bandwidth of the DMS. These are high in volume and place stringent demands on whatever communication network is available. Two options may be considered in determining how this traffic may best be handled. The first is to use all digital transmission and the space station DMS. The second is to use a separate analog network and retain the images in analog video form. Both options are viable and possess advantages and disadvantages. However, it should be noted that the choice of one or the other does not impact the functionality or operation of the VIP, but only affects the VNI.

4.3.2 Hardware Issues for VIP

Several distinct hardware issues arise in the organization of the VIP. These are related to the four principal components of the architecture: the VNI, the level-1 PEs, the interface between the level-1 and level-2 architectures, and the level-2 PEs. The functionality of the VNI and issues related to it have been discussed in the previous subsection. The EOSP architecture is an existing system, and most, if not all, hardware issues relevant to the VIP have been resolved. The operation of, and interface requirements to, an EOSP architecture are defined [3]. Issues related to the remaining two components are discussed in this subsection.

This interface is physically a bus that can accommodate data transfers at least at the sensor rate. Operation of this bus is embedded in the functionality of the VNI and the bus interface units of level-2 PEs. This bus, in addition to serving as the physical interface between the level-1 and level-2 architectures, also is the interface between the EOSP and the VNI for output of image data to the network. This bus is interfaced to the output buffers of the EOSP PEs. Since these buffers are externally clocked, some degree of freedom is available in designing the bus to interconnect the EOSP stages, the PEs at the next level, and the VNI. This sensor rate bus provides a parallel, multidrop data and message transfer medium and is a custom bus defined to meet the requirements of the VIP. The data transfer bus is a 16-bit parallel bus and thus is matched to the word width of the EOSP I/O data paths. The 16-bit bus also provides sufficient bandwidth for the anticipated data transfers. The control bus portion must provide signal lines for interrupts, bus arbitration, and broadcast. Given the block structured nature of data transfers, multicycle arbitration schemes with timeouts are probably preferable since the control overhead will be amortized over the size of the data transfers. In addition, with proper design of the flow of control, it is unlikely that all three components would be simultaneously requesting the bus. The interrupt facility would also be used to synchronize the transfers between the EOSP and the level-2 architecture. Use of a command facility for the sensor rate bus could eliminate the need for an address bus at the level-1 interface. The majority of data transfers across the sensor rate bus are block oriented rather than byte or word oriented. The EOSP output data is transferred across the sensor rate bus on a horizontal scan basis. Data transferred from the EOSP to the VNI is also based on the scan line as the unit of data transfer. A command code may be active during the beginning of a block transfer or for the duration of a transfer depending on the command type, e.g., beginning of a scan, EOSP microcode start address, end of scan, etc.

The two principal components of the level 2 architecture are the PEs and the multibus system interconnecting them. The architecture and its interface to the EOSP are illustrated in Figure 6. Each PE consists of a processor module with local memory, a global memory module, and bus interface units. The processor (with local memory) interfaces to the sensor rate bus and accesses the two inter-PE buses through the associated global memory element. Such an organization has several advantages. From the point of view of developing a testbed, all of the components and interfaces can be implemented with available standardized commercial components. This could actually continue to be the case, with some modifications, for a deployed version of the VIP. From a performance viewpoint, interspersing the processor between the global memory element and the sensor rate bus is crucial. This global memory element can provide performance equivalent to a locally accessible private memory for the local processor. At the same time, this element is available as globally accessible shared memory via the dual buses, and thus functions as a true shared memory since the local processor is not in the path for global memory accesses from remote PEs. The price paid for this generality is that the processor interface unit is in the path for data transfers from the EOSP, and the processor and memory share interfaces to the two inter-PE buses. Considering the synchronous, predictable, block structured nature of communication between the level-1 architecture and the PEs, this is not considered a significant disadvantage. Each memory element consists of fast access, static, random access memories (RAMs). Single-port access to the memory is provided by the local bus interface and the two level-2 global bus interfaces. All three bus interfaces would contend for port access on an equal priority basis.

Each PE consists of a processor, bus interface units, local bus systems, and a global memory element, as illustrated in Figure 6. The processor consists of a generic, 32-bit, single-chip computing element, such as the Motorola MC68030. Elements such as this can provide the computing power necessary to satisfy the throughput requirements determined by the VIP ADAS simulations. A complete set of software development tools, such as compilers, assemblers, and debuggers, is also typically available for such elements. The availability of such mature hardware/software environments is particularly advantageous for the testbed development phase.

The processor bus interface unit controls data transfers between the sensor rate bus, the PE and local memory, and the global memory element. Data may be transferred directly from the sensor rate bus to the global memory element. Data transfer may also occur between the local program memory and the global memory element. Each hardware node interfaces with a number of bus structures. The first is the sensor rate bus interface. The second is the intraprocessor bus system between the processor, local memory, and the bus interface unit. This would likely be a generic asynchronous bus interface well suited to interconnection between the generic processor and local memory. Finally, there is the local bus system between the processor unit and global memory element. A standard, 32-bit, asynchronous bus architecture, such as the VME bus [8], would suffice for this latter bus. An asynchronous bus structure for this local bus simplifies the bus protocol and allows for fast arbitration and capture of the system bus. This feature lowers PE dead time during a bus arbitration phase for single-word and short block transfers. Use of block transfers after the bus arbitration phase supports block-level direct memory access between the sensor rate bus and the global memory element.

A two-bus system architecture has been suggested for the VIP level-2 architecture. Global memory interconnection to the level-2 buses 1 and 2 is depicted in figure 6. There are a number of important qualities that the level-2 bus should possess. From the simulation studies, this bus system should provide a minimum average data transfer bandwidth of 5 Mbytes/sec. This performance figure is not difficult to achieve with many standardized bus architectures. A 32-bit data transfer bus width is preferred. This prevents packing and unpacking of 32-bit data that will be typically required. Further, the bus architecture should be processor independent and should allow a fairly large number of modules to interconnect to the buses. The current system calls for 16 PEs. In addition to the processor hardware nodes, there may be communications controllers that connect the VIP to the space station DMS via the level-2 bus.

A standard bus architecture that could meet the requrements for a VIP level-2 bus architecture is the Multibus II [9] bus structure. The Multibus II system bus is a high-performance, 32-bit bus capable of supporting up to 20 independent modules. This bus system is synchronous, supports processor independence, and supports block-level data transfers.

4.3.3 Software Issues

Currently under development is a microcode compiler for the EOSP and Distributed ADA [10], a candidate for the level 2 architecture. Capabilities have been successfully demonstrated on restricted problem sets. When finished they will enable the full VIP (level 1 and 2) to be programmed in ADA. With respect to operating system issues, we feel that a modification of an existing operating system such as Hunter and Ready's VRTX system will provide the functionality required of VIP. Schemes for distributed task allocation and scheduling are either handled within distributed ADA or have been developed [6]. Finally existing schemes are applicable for handling cache coherence and other problems that may arise. This is primarily due to the embedded nature of the VIP applications.

5 Concluding Remarks

Overall, a VIP will serve as a valuable utility to crew members on the space station, enabling them to efficiently accomplish their mission objectives and improve use of the space station resources, especially crew time. The architecture of VIP is based on relatively mature technology, one that will be stable before any future technology freeze date. Many of the systems issues can be resolved with existing hardware and software technology. The overall effect is one of comparatively low risk with the prospect of increased efficiency in many space station applications.

6 References

1. Honeywell Systems and Research Center, *Space Station Video Image Processor Concept Development*, Final Report, 1985.

2. P. Symosek et.al., *Knowledge-Based Vision for Spae Station Object Motion Detection, Recognition, and Tracking*, Proceedings of the NASA Workshop on Space Telerobotics, Pasadena CA., January 1987.

3. Honeywell Systems and Research Center, *Electro-Optical Signal Processor User Manual*, 1986.

4. B. Lint and T. K. Agerwala, *Communcation Issues in the Design and Analysis of Parallel Algorithms*, IEEE Transactions on Software Engineering, vol. SE-7, March 1981, pp.174-188.

5. S. H. Bokhari, *On the Mapping Problem*, IEEE Transactions on Computers, vol. C-30, March 1981, pp. 207-214.

6. Honeywell Systems and Research Center, *Video Image Processor for the Space Station*, Final Report, November 1986.

7. *Space Station Work Package 2 - Definition and Preliminary Design Plans*, Habitability/Man Systems Report, vol. 13, NAS9 - 17365 DR - 02, 1986.

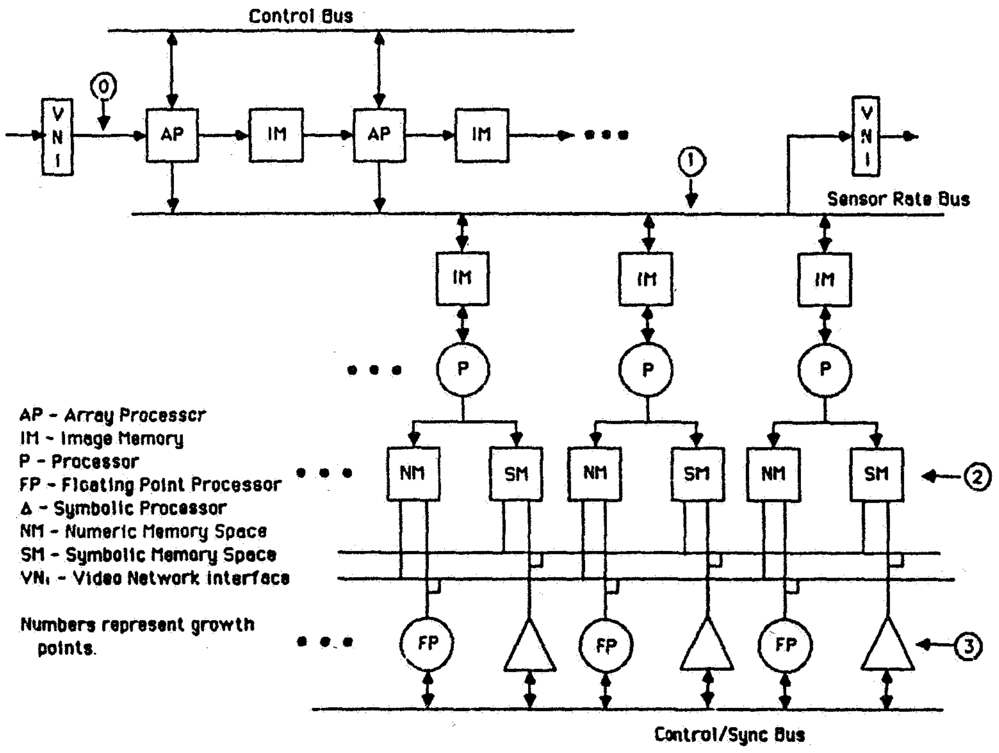
8. Signetics, VME Bus Manufacturers Group, *VME Bus Specification Manual*, rev. B, August 1982.

9. Intel, *Multibus II Bus Architecture Specification Handbook*, 1984.

10. Honeywell Systems and Research Center, *Honeywell Distributed ADA Project*, Status Report 1985.

	Color Image Enhancement	Tracking	Surveillance	Identification	Proximity Operations	Bandwidth Reduction
Construction	X	X		X	X	
Satellite Servicing	X	X		X	X	
Redezvous and Proximity Operations	X	X	X	X	X	
Inspection	X	X			X	X
Payload Delivery/Retrieval	X	X	X	X	X	X
Experiment Monitoring	X	X		X		X
Data Management and Communications	X	X		X		X
Training	X	X		X		X

Figure 1. Cross reference between applications and algorithms



Control Bus
0
VNI
AP
IM
AP
IM
1
VNI
Sensor Rate Bus
IM
P
NM
SM
2
FP
3
Control/Sync Bus
AP - Array Processor
IM - Image Memory
P - Processor
FP - Floating Point Processor
Δ - Symbolic Processor
NM - Numeric Memory Space
SM - Symbolic Memory Space
VNI - Video Network Interface
Numbers represent growth points.

Figure 2. VIP Conceptual Architecture

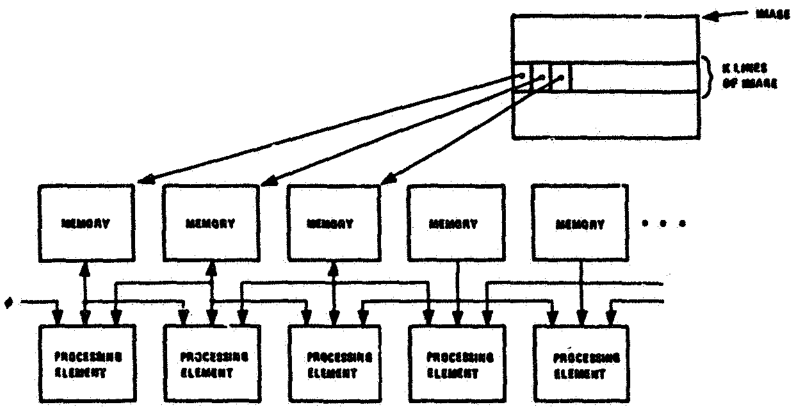


Figure 3. Organization of the EOSP

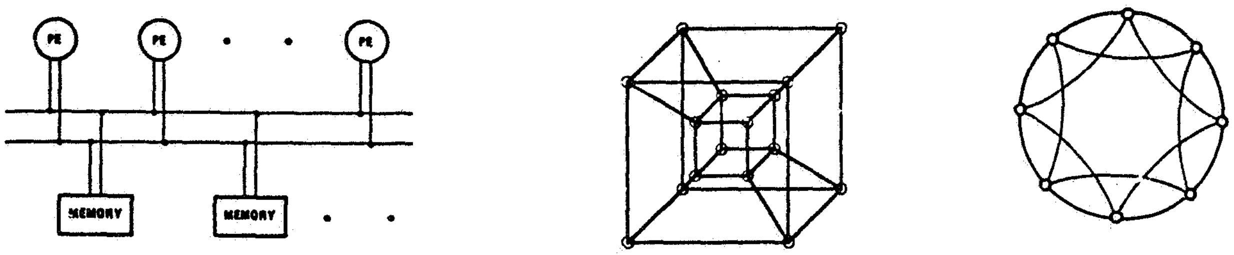


Figure 4. Bus oriented, hypercube and braided ring organizations

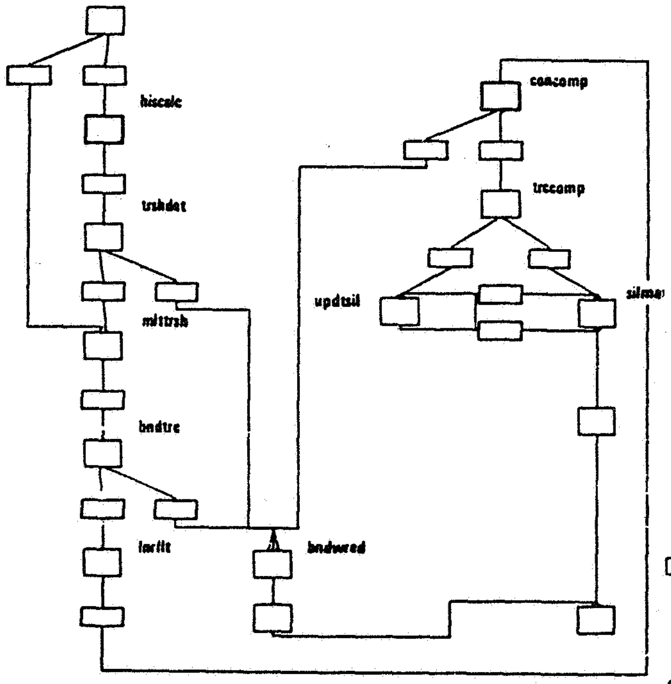


Figure 5. ADAS graph of the tracking and bandwidth reduction algorithm before parallelizing the components

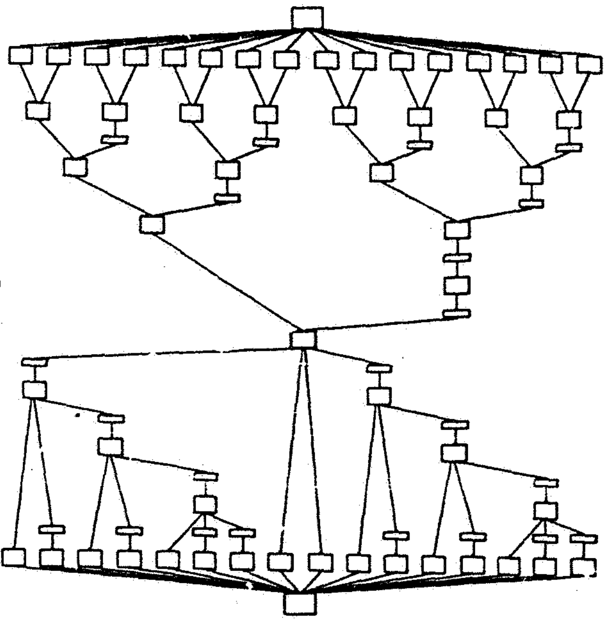


Figure 5 cont. Example ADAS software graph corresponding to a parallelized component

S12-74
78.

188132

An Optical Processor for Object Recognition and Tracking

PR 633222

J. Sloan and S. Udomkesmalee
Perkin-Elmer Corporation
Garden Grove, CA 92641

are described.

1. Abstract

~~This paper describes~~ the design and development of a miniaturized optical processor that performs real time image correlation. The optical correlator utilizes the Vander Lugt matched spatial filter technique. The correlation output, a focused beam of light, is imaged onto a CMOS photodetector array. In addition to performing target recognition, the device also tracks the target. The hardware, composed of optical and electro-optical components, occupies only 590 cm^3 of volume. A complete correlator system would also include an input imaging lens. This optical processing system is compact, rugged, requires only 3.5 watts of operating power, and weighs less than 3 kg. It represents a major achievement in miniaturizing optical processors. When considered as a special-purpose processing unit, it is an attractive alternative to conventional digital image recognition processing. It is conceivable that the combined technology of both optical and digital processing could result in a very advanced robot vision system.

2. Introduction

Coherent optical correlators have been successfully used in pattern recognition for years [1]. These correlators work on the familiar optical reconstruction property of holograms. The holograms are called Vander Lugt matched filters. A hologram of the target is made by simultaneously exposing the film plate to a reference beam and the 2-dimensional fourier transform of that target. Later, when the 2-dimensional fourier transform of a scene containing the target is imaged onto the hologram, it selectively re-directs the target's energy into a reconstructed reference beam which is focused onto a detector (see Fig. 1). The detector will see a "correlation spot" for every target in the scene, at a location that corresponds to that target's direction. The rest of the scene does not correlate, and appears essentially blank. This is an ideal format for position detection and video tracking.

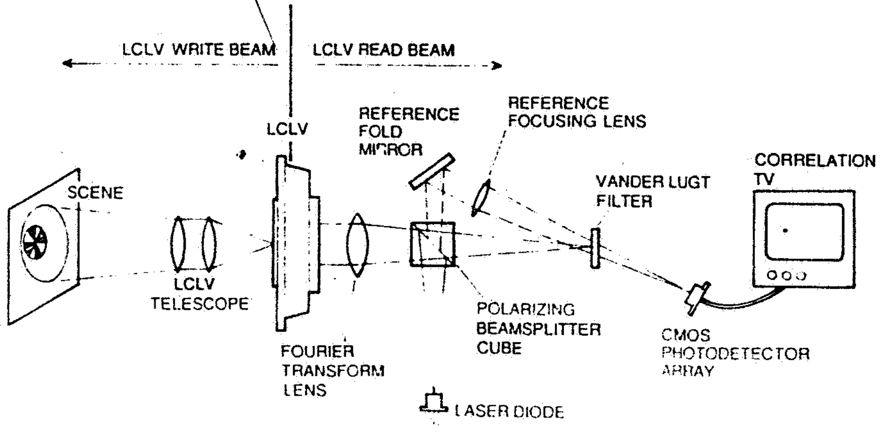


Fig. 1. The layout of the Solid Block Correlator. The correlator does not have a reference beam. When a target is correlated, the hologram reconstructs this beam, which was present in the HeNe system used to expose the hologram. The hologram focuses it onto a detector as a correlation spot.

The operation of optical correlators utilizing Vander Lugt matched spatial filters has thus far been primarily restricted to laboratory environments. In order to fully employ such correlators in real-world situations they need to be redesigned to survive field conditions, and deal with varying scene illuminations and realistic fields of view [2]. This paper presents the design and development of a miniaturized correlator that addresses these problems, and offers some future applications to which it is well suited.

The Perkin-Elmer Corp. has recently built the first of three optical correlation systems small enough to be gimballed inside a 150mm diameter missile (see Fig. 2). This Miniature Correlation Seeker System (MCSS) has been developed under contract to the Army's Missile Command (MICOM) to specifically demonstrate that an optical processor can be built rugged enough to operate inside a missile, and home it in on a pre-determined target. The correlator uses a Hughes Aircraft Co. Liquid Crystal Light Valve (LCLV) to convert the incoherent scene that the missile views to a coherent version suitable for optical processing by a Vander Lugt matched filter. The ability to immediately recognize and track a pre-selected target makes this system a viable option for navigation and guidance, docking maneuvers, robotic vision, and triangulation applications.

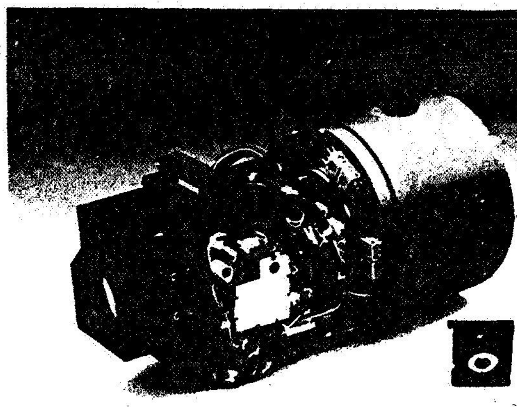


Fig. 2. Photograph of the Miniaturized Correlator Seeker Head, and a target's hologram.

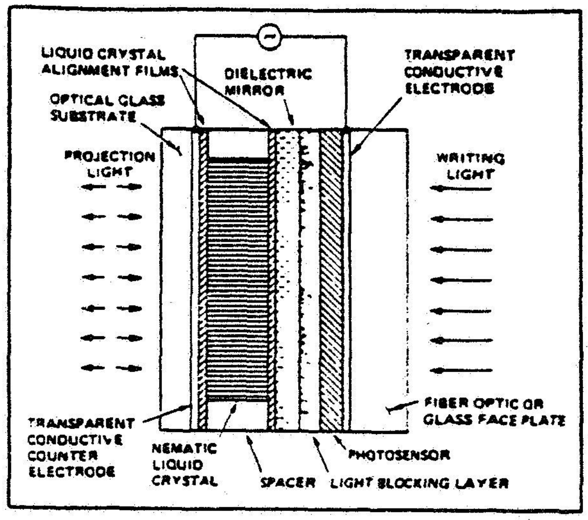
3. The Miniature Correlator Seeker System

The MCSS contains four key components: the Hughes LCLV, the Solid Block Correlator (SBC) module, the hologram, and a high-performance two-axis stabilized imaging platform.

The LCLV

The Hughes device is a hybrid field effect light valve utilizing the birefringence of the liquid crystal molecules, and their orientation, to modulate the polarization of the read light beam (see Fig. 3) [1]. The light valve is sensitive over a very narrow wavelength band on its write side (λ=520nm +/20 nm), and over a slightly broader band encompassing λ = 780nm, for reading. The difference in wavelengths, along with an internal light-blocking layer, effectively eliminates crosstalk between the two beams.

During operation of the light valve, an incoherent scene containing the target is imaged onto the write side of the LCLV with a telephoto lens. The variations in scene intensity locally alter the electric field permeating the liquid crystal layer which overlays the mirror on the LCLV's read side. The disturbed electric field changes the orientation of the liquid crystal molecules, thereby creating local rotations of the polarization of the read beam as it enters and exits the liquid crystal layer. Thus, the LCLV imprints an incoherent scene's intensities as rotations of the polarization of the independent read beam.



LIQUID CRYSTAL ALIGNMENT FILMS
DIELECTRIC MIRROR
TRANSPARENT CONDUCTIVE ELECTRODE
OPTICAL GLASS SUBSTRATE
PROJECTION LIGHT
WRITING LIGHT
TRANSPARENT CONDUCTIVE COUNTER ELECTRODE
NEMATIC LIQUID CRYSTAL
SPACER
LIGHT BLOCKING LAYER
PHOTOSENSOR
FIBER OPTIC OR GLASS FACE PLATE

Fig. 3. A cross-section of the Hughes Liquid Crysta Light Valve. The liquid crystal molecules are aligne in a bias AC electric field. When a scene is image onto the write side, the electric field changes, an re-orients the liquid crystal molecules. This locall alters the degree of rotation applied by the molecule to the polarized read laser beam. Thus, write imag intensities are converted to varying polarizations i a separate read beam. (Courtesy of Hughes)

The SBC Module

The SBC module, so called because the optical path is confined inside a folded glass prism assembly, fits in a cylindrical housing 100mm in diameter and 75mm high. The prism assembly is roughly octagonal in shape, and about 88mm across the corners (see Fig. 4). This solid configuration eliminates mounting problems, guarantees mirrors of good figure, and is easy to fabricate since all but two prism angles are 45°. Most importantly, alignments are permanent because the prism assembly is cemented together. Once the hologram is kinematically registered, it too is potted in place The prism assembly rides piggyback on the LCLV. A coherent light source, fourier transform lens, hologram correlation spot detector, and input imaging system for the LCLV complete the package (see Fig. 5).

A 30mW laser diode ($\lambda = 780$nm) is the coherent light source. Its polarized beam is apertured to reduce scattered light from its fan-shaped output. Spatial filtering is not required. A polarizing beam-splitter cube reflects the laser beam into the prism assembly. The beam is internally reflected around one half of the prism assembly by its 45° corners, and exits perpendicularly through the center of the assembly via a dove prism. There, a 2-element lens of 160mm focal length collimates the beam, which strikes the LCLV on its read side. The useful diameter of this read beam is 12mm. The LCLV modulates the polarization of the read beam according to the scene imaged on its write side. The reflected read beam then retraces its path through the collimating lens, which now acts as a fourier transform lens, and continues backwards through the prism assembly to the polarizing beam-splitter cube. Light that was not rotated by the LCLV is reflected back towards the laser diode. The rotated portions are transmitted by the beam-splitter cube to the hologram.

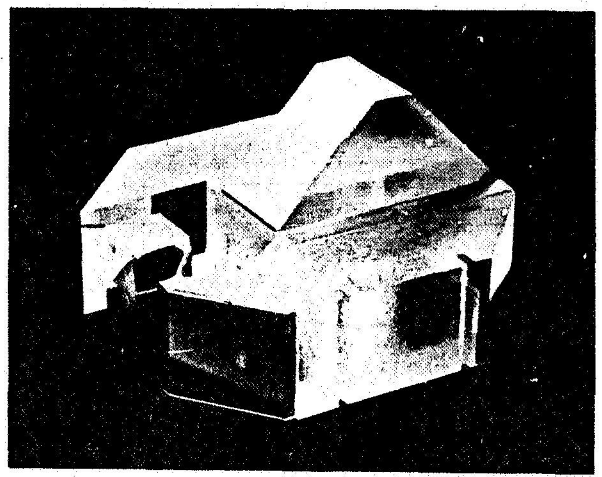


Fig. 4. Photograph of the Solid Block Correlator prism assembly. The prism assembly is 88mm across the corners.

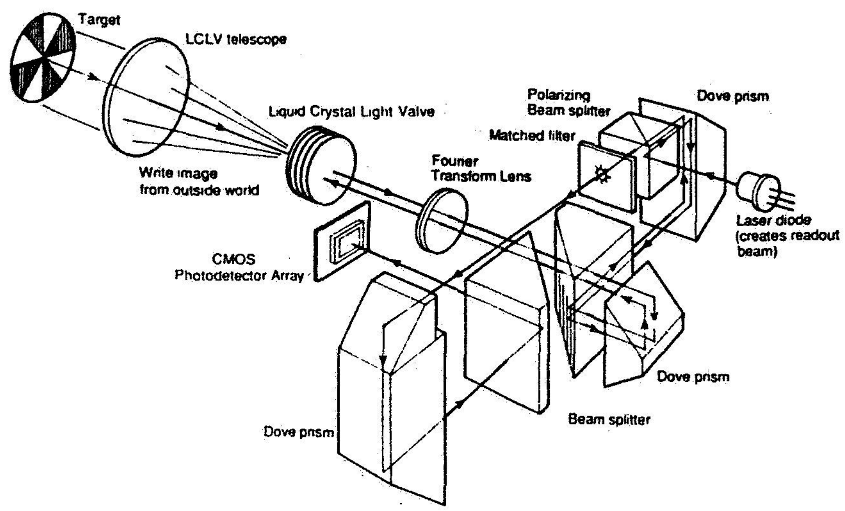


Fig. 5. Exploded view of the Solid Block Correlator module. The LCLV converts an incoherent white light scene into a version suitable for coherent optical processing. The fourier transform of this image is focused onto the Vander Lugt matched filter (hologram). If correlation occurs, the beam continues to the detector, a CMOS photodetector array, and appears as a focused spot. The location of the spot corresponds to the direction to the identified target.

The Hologram

The hologram of the target acts as both a Vander Lugt matched filter, and a holographic lens. A target's "matched" energy is diffracted and focused through the second half of the prism assembly to a detector, in this case a CMOS photodetector array. A target's correlation appears as a focused spot on the array at a position corresponding to the target's direction. If more than one target is in view at once, the correlator identifies and locates them simultaneously. With multiple exposures, the hologram becomes a multiplexed matched filter, capable of recognizing more than one view, scale, or type of target. The level of multiplexing depends on the method used. Overlayed holograms suffer from reduced efficiency, while separated holograms require multiple optical paths [3].

By using a laser diode, the SBC module is kept small. However, the photographic emulsion (Kodak 131-02) is not very sensitive at $\lambda = 780$nm, and a separate correlator system operating with a HeNe laser ($\lambda = 633$nm) is used to expose the holograms. (The laser diode can expose the film, given a long enough exposure, and this ability is used to great advantage to align the position of the hologram to the SBC's optical axis. The hologram's alignment is accurate to within a micron.) The HeNe system is precisely aberrated to compensate for the change in operating wavelengths, and is aligned to match the SBC's optical axis. A transparency of the selected target is placed in the collimated object beam, and its 2-dimensional fourier tran form, properly scaled, is imaged onto the 13mm diameter film plate. A reference beam, brought to a focus behind the film plate, is also required. This beam is the one reconstructed when the hologram finds a target correlation.

The ratio of intensities between the object and reference beams determines what details of the target will be used for correlation. To get high diffraction efficiency in a hologram, the local intensity in both beams should be equal at the film plane. When these two beams meet and interfere, fringes are formed in the film's emulsion. These create a diffraction grating at the locations of energy in the target's 2-dimensional fourier transform, or power spectrum. Very intense spatial frequencies in a target's power spectrum will over-expose the hologram, whereas very weak spatial frequencies will fail to interfere with the reference beam and won't produce fringes. Thus, only a restricted range of power spectrum intensities of a target can be adequately matched by the reference beam at any one exposure (see Fig. 6). For a target with a wide range of spatial frequencies, a determination of what features to use for correlation must be made, in keeping with the film's limitations.

Once the discriminating feature size is determined, the focal length of the LCLV's imaging system can be found. The correlating features must be magnified enough to be resolved by the LCLV. The MTF of the Hughes device extends out to 35 lp/mm, with 50% MTF around 20 lp/mm. The MCSS uses a 128mm f/4 lens system to image scenes onto the LCLV. This short focal length was in part determined by space restrictions inside the MICOM test missile, and a required field of view and field of regard.

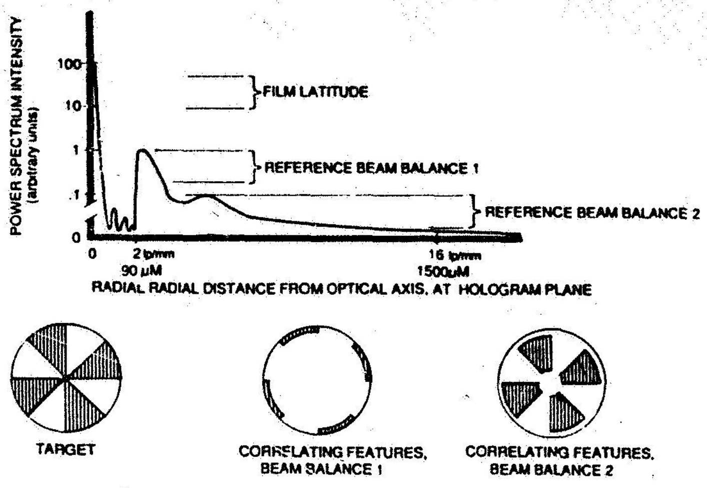


Fig. 6. A target, bottom left, and its power spectrum, top. The film's exposure latitude limits the range of intensities that the reference beam can match. If beam balance 1 is used, the filter will correlate against low frequency features, bottom middle. If beam balance 2 is used, mid-frequency features will correlate, bottom right. Beam balances must be chosen to correlate only on a target's distinguishing features, and yet be as generic as possible to accommodate different scales, angles of view, and changing aspect ratios.

The Stablized Platform

While the resolution of the LCLV is not affected by the write light intensity, its time response is highly dependent. Low write light levels ($30\mu W/cm^2$ at λ = 520nm) result in slow ($\geq$150 msec) rise or fall times depending on the driving frequency used to operate the light valve [1,4]. This makes it necessary to stabilize the input image on the LCLV. Even slight image motion will suppress fine detail or faint targets, rendering them invisible to the correlator. Until more responsive devices with the same resolution capability become available, a stablized platform will be a required adjunct for a dynamic optical correlator using a LCLV. An improved light valve could reduce the size of the MCSS to only 100mm in diameter and 200mm long. When mounted in a two-axis gimbal, the system becomes 150mm in diameter and 350mm in length. In order to keep the imaging platform aligned with the line-of-sight, rate integrating gyroscopes are mounted on each gimbal axis. These are coupled to a motor on the gimbal system such that a line-of-sight error command from a video tracker moves the platform at a rate proportional to the line-of-sight rate. The gimbal system has a pointing repeatability of 0.3 mrads, and a jitter stability of 10 μrads.

4. *Missile Application*

Presently, the MCSS will be used in a U.S. Army test program. It is a test of hardware concept only. The correlator will be mounted inside a missile, and perform target recognition as well as provide a guidance signal to direct the missile to its target. The missile is launched from a helicopter flying at an altitude of 5000 feet (see Fig. 7). The video tracker and guidance control computer are located at a ground command station and communicate with the missile via radio-frequency transmission channels. (The ground station is a hold-over from previous test programs. It would be possible to incorporate a guidance module in the missile, making the system completely autonomous.) When the target has been recognized (correlated), the missile is dropped from its carrier, but retains communication with the carrier through a fiber optic link. By continuously monitoring the rate integrating gyros, guidance information pertaining to the bearing of the line-of-sight can be directly obtained. Line-of-sight rates are transmitted to the missile steering system to maintain a constant-bearing course. The video tracker locates the correlation spot in the TV field by finding the peak pixel intensity. Correlation spot position errors are also transmitted to the rate integrating gyroscopes, enabling the seeker to track the target (see Fig. 8).

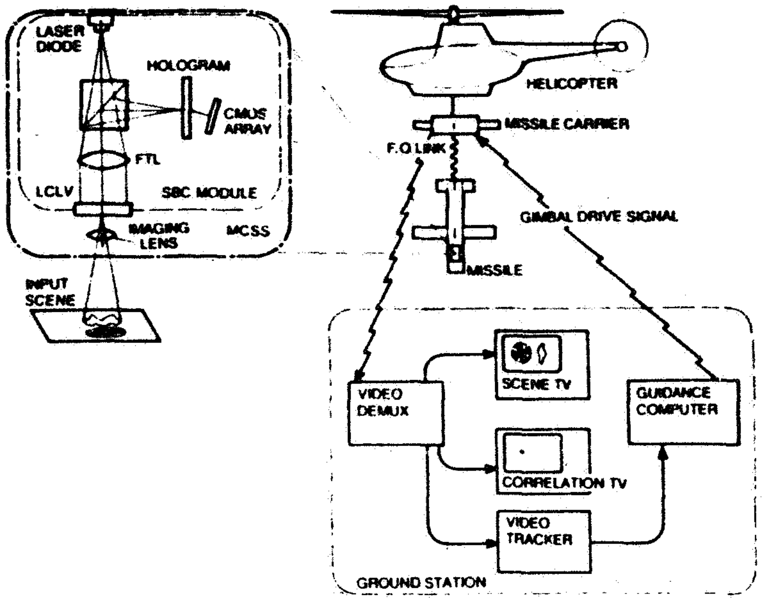


Fig. 7. A schematic view of the Army test program using the MCSS. The MCSS is installed in the front of the missile and will identify and track a specific target. Guidance commands will be generated from a ground control station, based on the information generated by the optical correlator.

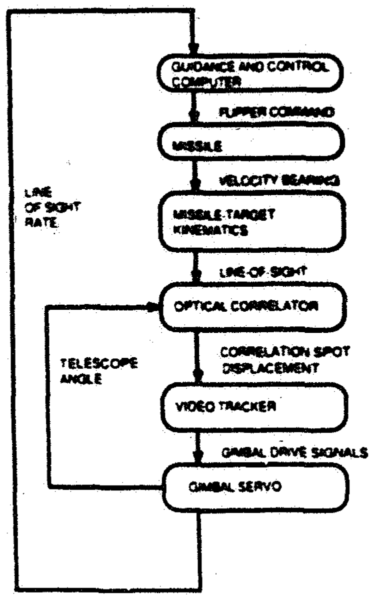


Fig. 8. A flow chart of the guidance loop used in the missile tests. The optical correlator allows the missile to find its own target without assistance from an operator.

5. Targets

In the Army's test program, the MCSS must track an approaching target over a 10:1 zoom ratio. This means that the spatial frequencies of the target also change scale by a factor of ten. A simple matched filter cannot correlate over this range unless some feature of the target is "invariant", i.e. unchanging in the spatial frequency domain. This includes not only the distance from the DC zero frequency, but the angular orientation too. For this reason, a special target is used. The MCSS uses a "cooperative" target which has this property. The target is a circle divided into 10° wedges, with 18 black sectors alternating with 18 white ones (see Fig. 9). As the target approaches the correlator, some mid-frequency detail will shrink out of the hologram's correlating zone while a higher frequency detail shrinks into that same zone. There is always some portion of the target that precisely matches the required correlation features. Additionally, a rotation of 10° has the effect of returning the pattern to its original orientation.

Unfortunately, a rotation of 5° is sufficient to displace the target's power spectrum at the hologram plane from the diffracting fringes, and correlation will not occur. This is remedied by double exposing the hologram so the correlator recognizes the original pattern as well as one rotated by 5°. In this manner, the "cooperative" target avoids the two problems of scale and orientation that plague optical correlators. (Instead of doubly exposing the hologram, the target could have twice the number of sectors. Now, a rotation of 5° restores the pattern to its original orientation and correlation is continuous throughout the rotation. However, this finer pattern would necessitate twice the magnification to the LCLV. This is impractical for the MCSS since the LCLV telescope is restricted by missile size. A minimum target image area is necessary to provide the SBC with enough reflected laser energy to correlate. The small image scale, coupled with the insensitivity of the LCLV, has already forced the target size to 10.66m in diameter.)

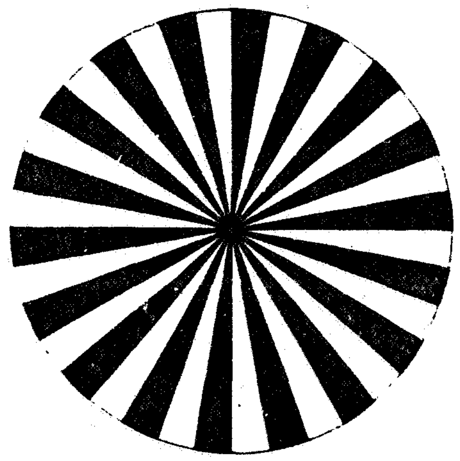


Fig. 9. The "cooperative" target used in the MICOM test program. This target is scale and orientation invariant, so it is easily correlated over a wide range of distances, and orientations.

The determination of optimum feature size for discrimination was found by experiment. The requirement for correlation at 5000 feet altitude meant the hologram had to be capable of correlating on fine detail. In this case, the hologram filter generator system was outfitted with a target transparency representing the angular subtense of the real target as viewed from 1/3 the maximum correlation distance. This is close to the geometric mean of the target's subtense over a correlation run. It represents a compromise of many parameters: energy available for correlation at maximum range (laser diode power per solid angle times the target image size on the LCLV times the sensitivity of the LCLV), emulsion exposure latitude, power spectrum energy per hologram area, number of correlating features of the target, and the quality of the generator's optical system. This compromise changes for different targets, including different views of the same object.

6. Future Space Applications

Although the difficulties in designing a scale and rotation insensitive optical correlator are not addressed in this paper, the MCSS can still perform useful machine sensing and perception functions. It can be used for object identification, docking maneuvers, and robotics, and is small enough to be used as a field unit. The correlator reduces a complex scene to a black background with a correlation spot, perfect for tracking computers and position analysis. There are some advantages to using "cooperative" targets instead of specific targets however.

A spacecraft with one "cooperative" target provides identification and directional information only (see Fig. 10). Two targets on a craft provides a twin spot correlation whose separation gives target distance information too. An asymmetric array of three targets provides relative angular orientation in addition. The same correlator can be used for any craft that carries the same cooperative target or target array. Targets can be used to guide a robot vehicle home as well. Multiple correlators can be used for accurate triangulation networks. In the current configuration, the MCSS can point to an accuracy of 0.2 mrads. as defined by the correlation CMOS photodetector pixel size. With pixel averaging, this can be improved significantly. These same principles can be used to guide a robot's arm to an object and pick it up. Targets need to be well illuminated, but not entirely visible to the correlator. With increased LCLV image scaling, target size can be markedly reduced, or detection range increased.

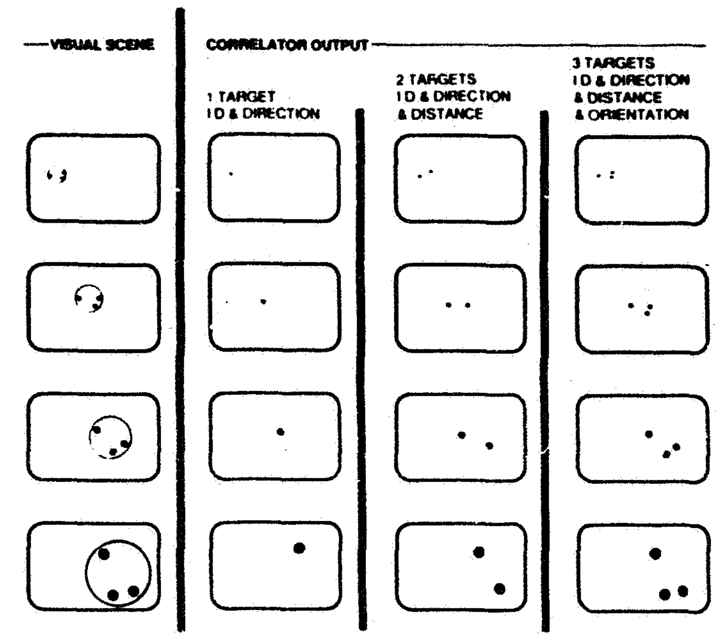


Fig. 10. Arrays of targets provide more information than single targets. A spacecraft, or single target, can be identified and tracked. Two targets will also yield distance data, and three will unambiguously resolve relative orientations.

7. Future Correlators

The SBC has limited "intelligence" in that it can only recognize those targets stored in its matched filter, some or all of which may be of the same object as viewed from different distances or directions. Experiments indicate that a maximum of about 30 targets can be stored on a 1 cm^2 hologram [3]. An additional handicap is the requirement that SBC holograms be made in a separate precision optical system. The future of the SBC lies in the incorporation of a larger target "memory", preferably one that can be actively manipulated and easily expanded.

One way to accomplish this would be to substitute a Programmable Spatial Light Modulator (PSLM) for the holographic matched filter. With a built-in EPROM of power spectra for various targets, and software to scale and rotate them, the SBC would become a much more versatile processor. A candidate PSLM is the Sight-Mod, a magneto-optical device manufactured by Semetex Inc. It uses core memory addressing to magnetically flip pixels from one optical polarity to another. An external analyzer converts the two states to transmissive or opaque (see Fig. 11). The erase/re-write cycle for the entire array is only a few milliseconds, and the array's state is non-volatile. Current devices have arrays of 128 x 128 pixels, with a pixel size of 76μm square. This coarse array would require a much longer focal length transform lens to scale the power spectrum to the large pixel size. A 3rd generation version with a 256 x 256 pixel array is in development. The Semetex package is relatively huge compared to the hologram in the SBC, with a size of 150mm x 150mm x 50mm thick.

Initial studies with the Sight-Mod have shown that it can perform simple optical processing [5]. More work is needed to explore its capabilities and limitations as a PSLM. The Semetex device is not the only one to consider, but the others also suffer from poor resolution or slow response times. There is no ideal PSLM yet, but the area is being actively researched. If progress follows the integrated circuit field, within a few years these devices will operate twice as fast, have twice the resolving capability, and use less power.

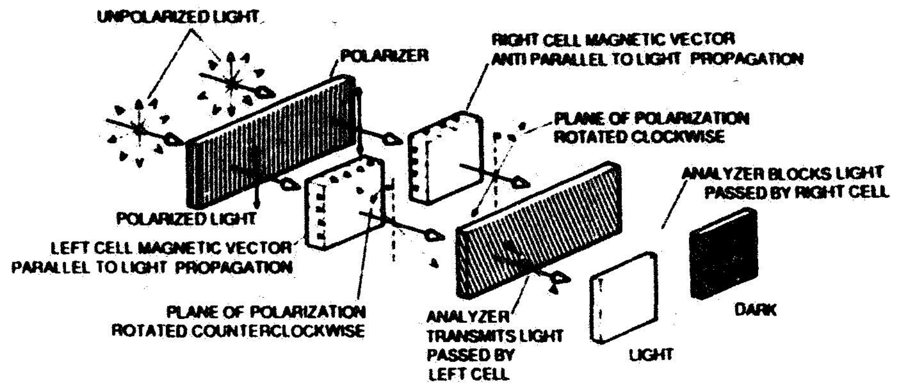

UNPOLARIZED LIGHT
POLARIZER
RIGHT CELL MAGNETIC VECTOR ANTI PARALLEL TO LIGHT PROPAGATION
PLANE OF POLARIZATION ROTATED CLOCKWISE
ANALYZER BLOCKS LIGHT PASSED BY RIGHT CELL
POLARIZED LIGHT
LEFT CELL MAGNETIC VECTOR PARALLEL TO LIGHT PROPAGATION
PLANE OF POLARIZATION ROTATED COUNTERCLOCKWISE
ANALYZER TRANSMITS LIGHT PASSED BY LEFT CELL
DARK
LIGHT

Fig. 11. Operation of the SIGHT-MOD as a light valve. Signals to the SIGHT-MOD array change the magnetic orientation of each pixel. Light passing through all of the "on" pixels is projected onto a display surface to create an image or character. (Courtesy of Semetex)

8. Conclusion

The miniaturization of a conerent optical correlator is an important achievement. It dismisses the idea that optical computers consist of a roomful of sensitively aligned optics and shows that they can be redesigned into rugged, independent units. The present missile application of this device demonstrates the feasibility of adapting image correlation technology towards real-world problems, but does not demonstrate the full potential. Scale and orientation changes severely limit the versatility of current optical processors. More work is needed to increase the memory capability of matched filters. The next generation of optical correlators may solve this problem with an electrically programmed hologram that has an easily manipulated and expandable memory of target spectra and quick time response.

Nevertheless, the SBC has immediate application opportunities in space and robotics. By using an array of three asymmetrically placed "cooperative" targets, information pertaining to identity, direction, distance, and orientation can be derived. Such capabilities are crucial for designing an autonomous robot system. Experience in using the correlator module in guidance and navigational applications, or robot vision systems, will prove invaluable for the design of future optical correlators.

9. References

[1] W. Bleha, et al., Opt. Eng., 17, 4, (1978)
[2] J. Duthie, J. Upatneiks, C. Christensen, R. McKenzie Jr., Proc SPIE 231, (1980)
[3] K. Leib, R. Bondurant, S. Hsiao, R. Wohlers, R. Herold, App. Opt., 17, 18, (1978)
[4] R. Buzzard, J. Sloan, Proc. SPIE, 684, (1986)
[5] D. Psaltis, F. Mok, E. Paek, Proc. SPIE, 465 (1984)

S/3-63

70.

188123

Improved Shape-Signature and Matching Methods for Model-Based Robotic Vision

J.T. Schwartz and H.J. Wolfson
New York University
New York, NY 10012

N0052345

[illegible]

1. Abstract

We describe new techniques for curve matching and model-based object recognition, which are based on the notion of shape-signature. The signature which we use is an approximation of pointwise curvature. The talk will describe a curve matching algorithm which generalizes a previous algorithm due to Schwartz and Sharir which was developed using this signature allowing improvement and generalization of a previous model-based object recognition scheme. The results and the experiments to be described relate to 2-D images. However natural extensions to the 3-D case exist and are being developed.

2. Introduction.

The purpose of this talk is to survey the work done in the Robotics Laboratory of the Courant Institute on *Model-Based Object Recognition* with emphasis on recent results. Recognition of industrial parts and their location in a factory environment is a major task in robot vision. Most industrial part recognition systems are model-based systems (see a survey in [1]). The model based approach is well suited for an industrial environment, since the objects processed by the robot are usually known in advance, and belong to a certain subset of the factory's tools and products.

We discuss the 2-D object recognition problem, where the robot is faced with a composite scene of overlapping parts (thus partially occluding each other), taken from a large data-base of known objects (e.g. the factory's warehouse). The task is to recognize the objects in the scene and their location. We want the recognition time to be fast and depend on the size of the scene, which is usually small, and not on the size of the initial large data-base.

The algorithms which we describe were actually tested in a "real life situation" by recognition of objects comprising composite scenes of about ten thin overlapping cardboard pieces taken from a data-base of hundred pieces. In our approach the camera is held at a constant height over the scene, i.e is suitable for a conveyor belt situation.

Since we are concerned with recognition of overlapping objects, we cannot make use of global features such as area, perimeter or centroid of a 2-D object. However, since a 2-D object is fully described by its boundary curve, both globally and locally, we can use these curves in our recognition process. This requires development of robust and efficient curve matching algorithms. These algorithms are applicable not only for object recognition tasks, but also to other tasks where curve matching is required.

3. Preprocessing and Data Acquisition

We begin with three major preprocessing steps :

1) Planar pieces are photographed by a black and white RCA 2000 camera, and the pictures are digitized and thresholded to get a binary image for each piece.
2) The boundary of each piece is extracted from the binary image. These boundary curves are our "experimental" curves.
3) A *smoothing* procedure is applied to each curve. We use the procedure which is described in detail in [2]. Basically, this expands the noisy curve to a narrow strip defined by a certain threshold value ϵ and then finds the shortest path lying in this 2ϵ-wide strip.(It may be imagined as stretching a loose rubber band within a narrow sleeve.) This gives a polygonal approximation of each observed curve.

4. The Schwartz-Sharir Curve Matching Algorithm

The first algorithm we are going to describe is due to Schwartz and Sharir (see [2]). Given a curve and a proper subcurve of it in the plane, it computes the rotation and translation of the subcurve relative to the curve which gives the best match in an L_2 kind of metric. Moreover it also computes the distance between the two aligned curves in this metric, thus giving us a score of the quality of the proposed match.

Take two curves C and C' in the plane and assume that C is a translated and rotated subcurve of C'. Both C and C' are assumed to have been smoothed (i.e. they are polygonal approximations of the original curves) and parametrized by arc length s. The matching we seek calls for determination of the offset s_0 and the Euclidean transformation E for which the curves $EC(s)$ and $C'(s+s_0)$ are closest to one another in the L^2 norm. Specifically, we represent each of the curves C, C' by a sequence of evenly spaced points on it, and let these sequences be $(\mathbf{u}_j)_{j=1}^{n}$ and $(\mathbf{v}_j)_{j=1}^{m}$ respectively. Assume first that both curves have the same starting point (i.e. $s_0=0$ and, hence, $m \geq n$). Matching amounts to finding a Euclidean motion E of the plane which will minimize the l^2 distance between the sequences $(E\mathbf{u}_j)_{j=1}^{n}$ and $(\mathbf{v}_j)_{j=1}^{n}$:

$$\Delta = \min_E \sum_{j=1}^{n} |E\mathbf{u}_j - \mathbf{v}_j|^2$$

To simplify the calculation, first translate C so that

$$\sum_{j=1}^{n} \mathbf{u}_j = 0$$

Next write E as $E\mathbf{u} = R_\theta \mathbf{u} + \mathbf{a}$, R_θ denoting a counterclockwise rotation by θ.
In such case, as it is shown in [S-S], the best match is obtained when

$$\mathbf{a} = \frac{1}{n} \sum_{j=1}^{n} \mathbf{v}_j$$

and θ is the negation of the polar angle of $\sum u_j \bar{v}_j$, where the vectors $\mathbf{u}_j$, $\mathbf{v}_j$ are regarded as complex numbers u_j, v_j. The least-square distance for this best match is given by

$$\Delta = \sum_{j=1}^{n} |\mathbf{v}_j|^2 - \frac{1}{n}|\sum_{j=1}^{n} \mathbf{v}_j|^2 + \sum_{j=1}^{n} |\mathbf{u}_j|^2 - 2|\sum_{j=1}^{n} u_j \bar{v}_j| \qquad (*)$$

If the curves do not have the same starting point, we have to match the sequence $(\mathbf{u}_j)_{j=1}^{n}$ to each of the contiguous subsequences $(\mathbf{v}_{j+d})_{j=1}^{n}$ of the sequence $(\mathbf{v}_j)_{j=1}^{m}$, for $d = 0, \ldots, m-n$.
For each such d (*) thus becomes

$$\Delta(d) = \sum_{j=d+1}^{d+n} |\mathbf{v}_j|^2 - \frac{1}{n}|\sum_{j=d+1}^{d+n} \mathbf{v}_j|^2 + \sum_{j=1}^{n} |\mathbf{u}_j|^2 - 2|\sum_{j=1}^{n} u_j \bar{v}_{j+d}|$$

We seek the minimum of the values $\Delta(d)$, $d = 0, \ldots, m-n$, which can be found in time $O(m \log m)$, using the fast Fourier transform algorithm for computing the convolutions $\sum_{j=1}^{n} u_j \bar{v}_{j+d}$.

The algorithm described above is a corner-stone of our subsequent techniques. It has two major advantages - speed and robustness, as demonstrated in an experiment of a successful computerized assembly of 100 piece jigsaw puzzles with a lot of almost similar pieces (see [3]). (The algorithm was used as the local curve matching algorithm between the boundary curves of the pieces providing scores to the goodness of this matches, and then a global algorithm, based on combinatorial optimization techniques, used this scores to compute the correct solution of the puzzle.)

The curve matching algorithm can be applied directly to give a first solution of the model-based object recognition problem. First we have to divide the boundary of our composite scene into subcurves, such that each such subcurve is supposedly part of the boundary curve of a different object in the scene. This can be done most simply by assuming that objects in a scene meet at sharp concave angles (the so called "breakpoints" in [4]). Then each such subcurve can be matched against the boundary curves of all the objects in our data-base to determine the best matching object. However, this approach has two serious drawbacks in regard to our original goals. First, the use of "breakpoints" may in some cases cause a wrong subdivision of the boundary of the composite scene; secondly, the speed of recognition grows linearly with the size of the data-base, indicating the need for a more efficient technique. Thus further development is required, and the following sections will describe solutions to these problems. However, as was mentioned before, this algorithm is an essential part of the later developed methods, mainly because of its robustness.

5. The Generalized Curve Matching Algorithm

In order to avoid the use of the "breakpoint" heuristic, and also be able to solve a number of other curve matching problems we require an efficient solution to the following more general curve matching problem:

given two curves, find the longest matching subcurve which appears in both curves.

An approach to this problem due to Wolfson (see [6]) can be summarized as follows :

Step A : Represent both curves by *characteristic strings* which represent local translationally and rotationally invariant features.

Step B : Find the longest common substring of the two characteristic strings, and also find other long common substrings if these are nearly as long.

Step C : For each substring produced by *Step B* , go back to the original curves and match the two subcurves which correspond to this substring using the preceding subcurve matching algorithm, thus determining the desired translation and rotation of one curve with respect to the other.

Step D : Rotate and translate the curves accordingly, and determine again the longest matching subcurves of the two curves, given this rotation and translation. This subcurve is found by simply checking the (x,y) coordinates of corresponding points on the curves and demanding that the distance between the points should be less than a certain threshold value ϵ. This final check works with points on the curves themselves, and not with the (less accurate) feature string values at these points; hence it is quite robust.

The result giving the longest matching subcurve (allowing minor mismatches) is chosen as the final solution.

Two algorithms using this approach were developed, reflecting a certain tradeoff between robustness and the theoretical complexity of these proposed techniques. One of them uses efficient string matching techniques due to Weiner and to McCreight (see [7],[8]) to find the long matching substrings in time which is linear in the length of the strings. This makes the whole algorithm linear in the number of sample points on both curves, since the information obtained at *Step B* allows the curve matching algorithm to be implemented in linear time as well. However, the string matching algorithms mentioned above require the string elements with which they work to be taken from a finite alphabet, which forces us to truncate the feature strings (which consist of real numbers). This may cause otherwise long matching substrings to split into a number of shorter matching strings. To overcome this problem we developed a string matching algorithm which regards string elements equal when the difference between the two elements is less than a certain threshold value ϵ. This algorithm can be implemented in time which is proportional to $Max(n \log n, \epsilon n^2)$, thus making this algorithm quite efficient for curves of practical length. Experiments have shown this algorithm to be both efficient and robust.

This approach enables us to solve the object recognition problem by matching the boundary curve of the composite scene against the boundary curves of the objects in the data-base. Objects having long subcurves matching the composite scene and satisfying obvious consistency requirements will be objects participating in the scene. However this approach still does not satisfy the efficiency goal that we have set, since it is linearly dependent on the number of objects in the data-base. Thus additional ideas must be applied to the solution of the object recognition problem.

6. Shape Signatures

The method described in the previous section uses local rotationally and translationally invariant features to characterize boundary curves. In this section we will examine one such feature.

Our aim is to represent any curve C by a characteristic string of reals $(c_i)_{i=1}^n$. Since these strings will be compared to achieve subcurve matching, we want the numbers $(c_i)_{i=1}^n$ to encode characteristics of the curve which are :

i) local,

ii) translationally and rotationally invariant,

iii) stable, in the sense that small changes in the curve induce small effects (or no effect at all) in the associated sequence $(c_i)_{i=1}^n$,

a further desirable, but less essential, property is :

iv) an approximation to an observed curve can be reconstructed from its characteristic string.

One "natural" feature which satisfies these conditions is the pointwise *curvature* of a curve (see Chapter II of [9]). It is well known that there is a one to one correspondence between a regular curve (modulo translation and rotation) and its curvature function (which is a continuous function of its arclength). However, our applications must deal with noisy polygonal representations of curves, making it impossible to compute curvatures either accurately, or at every point of a curve. Thus we must work with an approximation of the curvature, calculated at discrete points of the curve, to get a data sequence $(c_i)_{i=1}^n$ having the desired properties.

Let $\kappa(s)$ be the curvature function of a curve C, where s denotes arclength along the curve. $\kappa(s)$ is the derivative of the tangent angle $\theta(s)$ to the curve, parametrized as a function of its arclength. To approximate the curvature, we first build the so called *arclength versus turning angle* graph of the curve C. (Since after our smoothing procedure we have a polygonal approximation of the observed curve, this is a step function.) Then we sample this graph at equally spaced points, and at every such point s_i ($i = 1,\ldots,n$) we compute the difference

$$\Delta\theta(s_i) = \theta(s_i + \Delta s) - \theta(s_i)$$

(To make the method more robust we actually compute an averaged difference

$$\phi_i = Av\Delta\theta(s_i) = \frac{1}{k}\sum_{j=0}^{k-1} \Delta\theta(s_i + j\delta)$$

Detailed choice of the parameters Δs, k, δ is based on experimental considerations.)

Remark: The averaged differences (ϕ_i) satisfy the conditions (i)-(iii) required of local curve characteristics.

At first glance algorithms based on such features would not seem to be robust, since we are computing approximations of a second derivative of an initially noisy function. Thus this kind of signature needs to be applied with care, mainly in preliminary steps which aim simply to filter out obvious "wrong" candidates in an efficient way, to prepare for final decisions made using more robust procedures. Note that in the previous algorithms described the use of the feature strings was quite limited: we used it to locate approximate starting points and endpoints of several long candidate subcurves for matching. Matching itself is done using the robust subcurve matching algorithm.

In the following section we will describe another use of shape signatures which solves the object recognition problem in a still more efficient way.

7. The "Footprint" Approach

In this section we give a method which solves the object recognition problem in a particularly efficient way. The method was first developed by Kalvin, Schonberg, Schwartz and Sharir in [4] and later improved by Hong and Wolfson (see [5]). We will present the later version, which differs from the previous one in two aspects - it does not require use of "breakpoints" in advance to divide the boundary curve of the composite scene into subcurves belonging to different objects, and it uses shape signatures based on approximate local curvatures, rather than the Fourier descriptors used in the previous version.

The algorithm consists of two major steps. The first one is a preprocessing step which is done on the data-base of model objects to be recognized. The complexity of this step is linear in the size of the data-base. This step can be executed off-line before actual recognition is needed. The second step, recognition proper, uses the data prepared by the first step and can be executed in time which on the average is linearly dependent on the size of the composite scene, thus achieving recognition time almost independent on the size and number of objects in the data-base.

A) Preprocessing

All the objects in the data-base are processed as follows. The boundary curve of every object is scanned and shape signature values are generated at equally spaced points. (We use a 5-tuple of approximate local curvatures at consecutive points.) This "footprint" is local, translationally and rotationally invariant. For each such footprint we record the object number and the sample point number at which this footprint was generated. (This data is held as a hash-table.) The "footprint" data points on the boundary of the object have a natural order, which is defined by the way the boundary curve is traced. This preprocessing step is linearly dependent on the total of sample points on the boundary curves of all the objects in the data-base. New objects added to the data-base can be processed independently without recomputing the hash-table (except when we must change its size by rehashing).

B) Recognition

In the recognition stage the boundary curve of the composite scene is scanned and footprints are computed at equally spaced points. For each such footprint we check the appropriate entry in the hash-table, and for every pair of object number and sample point number, which appears there, we tally a vote for the object and the relative shift between the object and the scene. For example, if a footprint, which was computed at the i'th sample point on the composite scene, appeared on objects k_1 and k_2 at sample points j_1 and j_2 respectively, we add votes to object k_1 with relative shift $i-j_1$ and object k_2 with relative shift $i-j_2$.

At the end of this process we find those (object,shift) pairs that got most of the votes, and for every such pair determine approximate starting and endpoints of match between the footprint string of the composite scene and the footprint string of the object under the appropriate shift. Given these matching substrings we may apply the robust subcurve matching algorithm described previously. This process resembles that used in the generalized curve matching algorithm presented in section 5. Once the object which has the longest matching subcurve with the composite scene is discovered, we decide that it is one of the objects in the scene, discard its matching subcurve, and repeat the process for the remaining curves in the reduced composite scene. At this stage a number of objects can be processed simultaneously.

The algorithm described is on the average linear in the number of sample points in the composite scene, and does not depend directly on the number of points on the boundaries of all the objects in the data-base. Thus we achieve the efficiency goal set at the beginning.

An improvement of this method can be achieved by the introduction of "weighted" footprints. Since for typical curves in different environments not all the footprints have an equal probability of occurrence, it seems desirable not to give an equal weight to every "hit", but to give a higher weight to coincidence of "rare" footprints. The actual probability of an individual footprint can be estimated by the number of its occurrences in the data-base, which can serve as a statistical sample for this kind of data. The weighted footprint approach can also improve its efficiency by assigning zero weight to very frequent footprints and thus saving us the need to process hash-table entries with a lot of candidates. These entries require much computer time but contribute only a small amount of information.

The method described above has proved to be quite robust. It also generalizes previously used methods based on use of special boundary features such as sharp angles. In our approach this is a special case, these special features being assigned large weight, while other features get zero weight. An appropriately sophisticated weight function can benefit from all the available information, and can deal with scenes which have no sharp angles or any other distinctive features, which are known in advance.

A major potential advantage of our "footprint" algorithm is its high inherent parallelism. Parallel implementation of this algorithm is straightforward; moreover, it should be quite easy to build a special device for this implementing it at very high speed.

8. 3-D Curve Matching

All the algorithms described here apply just as well to 3-D curve matching. The subcurve matching algorithm of section 3 has been implemented in the 3-D case (see [10]) and seen to perform well; currently a generalized curve matching algorithm, which uses 3-D shape signatures is being implemented.

9. Acknowledgements

Work on this paper was supported by Office of Naval Research Grant N00014-82-K-0381 and National Science Foundation Grant No. NSF-DCR-83-20085.

REFERENCES

[1] R.T. Chin and C.R. Dyer, *Model-Based Recognition in Robot Vision*, Computing Surveys, Vol. 18, No.1, March 1986, pp.67-108.

[2] J.T. Schwartz, and M. Sharir, *Identification of Partially Obscured Objects in Two Dimensions by Matching of Noisy "Characteristic Curves"*. Tech. Rep. No.165, June 1985, Comp. Sc. Div., Courant Inst. of Math., NYU (to appear in the Int. J. of Robotics Research).

[3] A. Kalvin, Y. Lamdan,E. Schonberg, and H. Wolfson, *Solving Jigsaw Puzzles Using Computer Vision*, Tech. Rep. No.205, Feb. 1986, Comp. Sc. Div., Courant Inst. of Math., NYU (to appear in "Approaches to Intelligent Decision Support" in the series Annals of Operations Research (1987))

[4] A. Kalvin, E. Schonberg, J.T. Schwartz, and M. Sharir, *Two Dimensional Model Based Boundary Matching Using Footprints*, Tech. Rep. No.162, May 1985, Comp. Sc. Div., Courant Inst. of Math., NYU (to appear in the Int. J. of

Robotics Research 5(4) (1986)).

[5] J.W. Hong and H. Wolfson, *On Footprints,* in preparation.

[6] H. Wolfson, *On Curve Matching,* Tech. Rep. No.256, Nov. 1986, Comp. Sc. Div., Courant Inst. of Math., NYU.

[7] P. Weiner, *Linear Pattern Matching Algorithms* , 14'th Annual Sym. on Switching & Automata Th.(Iowa), IEEE Comp. Soc.,(1973), pp. 1-11.

[8] M. McCreight, *A Space-Economical Suffix Tree Construction Algorithm,* J. of the ACM, Vol.23, No.2, April 1976, pp. 262-272.

[9] J.J. Stoker, *Differential Geometry,* Wiley-Interscience, 1969.

[10] M. Bastuscheck, E. Schonberg, J.T. Schwartz, and M. Sharir, *Object Recognition by 3-Dimensional Curve Matching,* Tech. Rep. No.203, Jan. 1986, Comp. Sc. Div., Courant Inst. of Math., NYU (Int. J. of Intelligent Systems (1986)).

S4-63

130

188134

A Technique for 3-D Robot Vision for Space Applications[1]

V. Markandey, H. Tagare, and R.J.P. deFigueiredo
Rice University
Houston, TX 77251-1892

RV 347060

1) ABSTRACT:

~~This paper reports~~ an extension of the MIAG algorithm for recognition and motion parameter determination of general 3D polyhedral objects based on model matching techniques and using *Moment Invariants* as features of object representation. Results of tests conducted on the algorithm under conditions simulating space conditions are presented. is discussed.

2) INTRODUCTION:

Many different object recognition and attitude determination techniques have been proposed by researchers. The earliest ones used the approach of matching the observed image to a library of a fixed number of views of objects. The limitations of such an approach are glaringly apparent. Among the later techniques, Richard and Hemami [1] used Fourier descriptors and Dudani et al [2] used moment invariants. Watson and Shapiro [3] used a model matching technique to identify wireframe perspective views. Their method is iterative and requires use of a numerical optimization technique. Marr and Poggio [4] have implemented a stereo reconstruction algorithm which uses geometric constraints to recover surface shape. Similar range data techniques have been developed by other researchers also. A fundamental limitation of these techniques is the introduction of restrictive assumptions about the imaged scene in terms of generalized cones [5] or in terms of planar and quadric patches [6]. Horn [7] has worked on the extraction of shape from shading, using the reflectance map. This method uses the brightness gradient as the image feature used in recovery. It is applicable to smooth, uniform Lambertian surfaces. Stevens [8], Kender [9] and later Witkin [10] have tried to recover shape from texture. This technique and also the shape from contour (surface boundaries) technique presented by Barrow and Tenenbaum [11] rely on the assumption that the world of objects is regular. Such techniques are limited to smooth-textured surfaces. For some other contributions see Silcox [12].

Bamieh and deFigueiredo [12] have developed the Moment-Invariants / Attributed Graph (MIAG) Algorithm in which 2D moment invariants which are invariant under 3D motion, have been used for the recognition of 3D objects, using an attributed graph representation and based on the concept of model matching. This approach avoids restrictive geometric assumptions and so offers an advantage over most techniques discussed above. In its original form this algorithm was applicable for recognition of polyhedral objects but it could not be used for attitude determination if the polyhedron had symmetric faces for reasons discussed later in this paper. This limitation has been overcome now as discussed in this paper. As this technique uses moment invariants as features of representation and these can be computed only for planar faces, the technique is not directly applicable to the recognition and attitude determination of curved objects. However we can use other representational features such as the Gaussian and mean curvature and the attributed graph and model matching techniques can still be applied.

To implement this technique we need picture information in the form of wireframes. So the picture from the camera is digitized and converted to wireframe form before applying the MIAG-algorithm to it. In the work currently in progress, the overall process is divided into three parts:

[1]Supported by the NASA contract NAS 9-17145, the NASA/JSC grant NCC 9-16, and the NSF grant DCR-8318514.

1) Data acquisition and digitization.

2) Wireframe extraction.

3) Recognition and motion parameter determination.

The first two parts above constitute the image processing/feature extraction stage. The work in this stage is briefly outlined below.

3) IMAGE PROCESSING / FEATURE EXTRACTION:

3.1) Data acquisition and digitization: Models of various space objects, such as mockups of satellites, the space shuttle and parts of the space station are being used to test the performance of the algorithm. These models are grabbed by cameras under illumination conditions simulating those prevelent in space orbit. The pictures are digitized to obtain 2D arrays of brightness values. This is the initial level of representation in the system.

3.2) Wireframe extraction: Wireframe extraction consists of removing noise from the picture and subjecting it to edge detection and reconstruction. The input image is lowpass filtered to remove the high frequency noise. A 7x7 Gaussian filter is used for this. The Sobel gradient operator is then applied to the output of the lowpass filter to obtain an edge detected version of the input image. This image is a grey level image. It is converted to binary form by thresholding, which also removes some of the noise and thins down the edges. The remaining noise is removed by median filtering. A length 5 filter was employed for this. The output so obtained is a noise free, binary edge image. But the edges are thick smears instead of the fine lines required in a wireframe. A thinning algorithm [13] is applied to this to reduce the edges to unit pixel thickness, thus obtaining the required wireframe.

4) RECOGNITION AND MOTION PARAMETER DETERMINATION:

The MIAG algorithm [12] is an algorithm of recognition and attitude determination of 3D objects. We discuss this algorithm in two steps: First, object recognition and second, attitude determination.

4.1) Object recognition: The MIAG technique recognizes a 3D object from its projection on an imaging plane. The algorithm works for the identification of polyhedral objects. Each face of a polyhedron can be considered as a rigid planar patch (RPP). Motion of the object can then be considered as motion of its constituent RPP's. If it is assumed that the image is formed by parallel projection then if an RPP undergoes rigid body motion in 3D its image undergoes affine transformations. So the method which tries to identify an object in 3D motion should use features of images which remain invariant under affine transformations. General moment invariants are such features. They remain invariant under translation, rotation and scale changing. Moments are coefficients in a series expansion of the image function, similar to those in a Fourier series expansion. But unlike in Fourier series where sine and cosine functions are the basis functions, here the basis functions are polynomials in the image function variables. Thus if the picture function is f(x,y) its moment is:

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q f(x,y)\,dx\,dy$$

for p,q=0,1,2...

The value of (p+q) is known as the order of the moment. Theoretically, for a perfect description of the picture in terms of moments, p and q should go to ∞. But in the present algorithm moments upto only order four have been used. This is because the computation of higher order moments is increasingly difficult, and it was found that picture representation in terms of four moments gives good results in the test cases.

These moments have to be computed for each face of the picture wireframe. The picture intensity is taken to be 1 inside a polygon and 0 outside it. Thus all the picture information is contained in its boundaries. Using this fact, the above surface integral can be changed to a line integral by Green's theorem. For a digital picture the integral reduces to a sum. See [12] for details.

Moment invariants of all faces of certain standard objects are stored in the system library. Given a wireframe which needs to be identified, the moment invariants of each of its faces are computed. These are then matched to the stored values of the moment invariants of the library objects. If all the moments corresponding to a face of the RPP match all the moments of a face of a stored object, we can say that the two faces are similar. For the objects to be the same, not only should the faces be similar but the adjacency conditions of the faces and the angles between the faces should be similar. To carry out this matching, the wireframe is first converted to an attributed graph. Each node of the graph represents a face of the wireframe. If two nodes are connected by a line (edge) it means that the faces corresponding to these nodes are adjacent. With each node is associated a feature vector consisting of a set of moment invariants of the face that it represents and with each edge is associated a scalar which gives the angle between normals to the two faces it connects. As an example, the attributed graph representation of a cube is shown in Fig 1.

The algorithm works as follows: Suppose we hypothesize that node W_j in the wireframe corresponds to node O_j in the model graph. If W_j has k nodes adjacent to it, $W_{m1},...,W_{mk}$ then O_j should also have k nodes adjacent to it, $O_{n1},...,O_{nk}$. The following constraints should be satisfied:

1) W_{ms} must have the same feature vectors as O_{ns},s=1,...,k.

2) Angle between W_{ms} and W_j should equal angle between O_{ns} and O_j for all s=1,...,k.

3) If any two W_{ms} 's are connected then the angle between them should equal that between their matching nodes.

If all these conditions are satisfied, then an admissible matching configuration is said to have been obtained at nodes W_j and O_j. If matching configurations are obtained between all nodes of the given wireframe and one of the stored models, then we can say that the wireframe matches the model. In most cases, only a small part of the given model needs to be matched to discriminate it against the other models in the library.

It may be noted that because of numerical truncations and rounding during calculations there may not be a perfect match between the moment invariants computed for the wireframe and those stored for the model. So we define a measure of error between the two sets of moment invariants. The moment invariants can be taken as coordinates of a point in four dimensioinal vector space and the distance between the two points is taken as a measure of error. If I_1, I_2, I_3 and I_4 are the moment invariants of a wireframe's face and I'_1, I'_2, I'_3 and I_4 those of a model's face then the distance is:

$$d = \sqrt{(I_1 - I'_1)^2\rho_1^2 + (I_2 - I'_2)^2\rho_2^2 + (I_3 - I'_3)^2\rho_3^2 + (I_4 - I'_4)^2\rho_4^2}$$

where the ρ's are weighting factors. These are needed to equalize the contribution of all four moment invariants in the error measure because some of the moment invariants may have values of the order of 10^{-3} and others may be of the order of 10^{-7}. If the value 'd' is less than a certain threshold (taken 0.01 here), the two sets of moment invariants are taken to be equivalent.

The driver algorithm arbitrarily picks a node W_j in the wireframe, then it looks for a node O_j in the model with the same feature vector. If matched, these nodes are marked as a pair. An adjacent image face is chosen and the adjacent object faces are scanned to see if one of them matches it. As each adjacent pair is found it is checked for consistent adjacency and equality of angles between faces. If everything matches satisfactorily a succesful match is declared.

4.2) Attitude determination: The identity of the object having been so determined, one has to estimate the attitude and location of the recognized object relative to a library standard.

Let (X,Y,Z) be the original coordinates of a point on the body and (X',Y',Z') be its coordinates after motion. Then,

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + T$$

where

$$R = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_8 \end{bmatrix}$$

is the rotation matrix and

$$T = \begin{bmatrix} \Delta X \\ \Delta Y \\ \Delta Z \end{bmatrix}$$

is the translation matrix.

Let the corresponding points on the image be (x,y) and (x',y'). Then,

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} r_1x+r_2y+r_3Z+\Delta x \\ r_4x+r_5y+r_6Z+\Delta y \end{bmatrix}$$

For simplicity let Z=0 i.e. the RPP in library lies in the x-y plane of the object space. Then the above matrix can be represented as:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} Q_1^1 & Q_2^1 \\ Q_1^2 & Q_2^2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \Delta X \\ \Delta Y \end{bmatrix}$$

where

$Q_1^1 = r_1, Q_2^1 = r_2, Q_1^2 = r_4, Q_2^2 = r_5, \Delta x = \Delta X, \Delta y = \Delta Y.$

All Q's and Δ's can be determined from the moments of the images. To find the rest of the r's we use the fact that the sum of squares of any row or column in the r matrix is 1.

From the r's we can find directional cosines of the rotation axis and the rotation angle as:

$$\sin\theta = \frac{d}{2} \text{ and } \cos\theta = \frac{d^2 r_1 - (r_8 - r_6^2)}{d^2 - (r_8 - r_6^2)}$$

where

$$d = \sqrt{(r_8 - r_6)^2 + (r_3 - r_7)^2 + (r_4 - r_2)^2}$$

(both sin θ and cos θ are needed to determine θ uniquely). The direction cosines are:

n1 = $(r_8 - r_6) / d$

n2 = $(r_3 - r_7) / d$

n3 = $(r_4 - r_2) / d$

where $d^2 = (r8-r6)^2 + (r3-r7)^2 + (r4-r2)^2$

We can also find the translation in x and y directions as:

$\Delta x = m_{10} / m_{00}$; $\Delta y = m_{01} / m_{00}$

Δ z is not computable by this method. Also, this method cannot give rotational information for objects which have an axis of reflection symmetry (e.g. parallelograms, triangles) as the tensors all go to zero in such cases. So given an RPP whose attitude has to be determined we need to:

a) Check whether any axis of reflection symmetry exists.

b) Check whether it will have any axis of reflection symmetry under any affine transformation.

c) If the face has any axis of reflection symmetry or will have it under affine transformations, subject it to distortion which removes the axes of reflection symmetry.

The procedures for these steps are as follows:

a) Symmetry conditions for polygons: To check a given polygon for axes of reflection symmetry, we use the concept of the *Voronoi diagram.*

4.2.1) Voronoi diagram: Given a set of N points corresponding to the vertices of the polygon, let x^i and x^j be two of the points. Let P(x^i , x^j) be the half plane containing x^i that is defined by the perpendicular bisector of x^i x^j . The intersection of N - 1 such half planes, denoted by V(i) is called the Voronoi polygon associated with p^i . Note that the polygons are unbounded. For N points there are N such polygons which partition the plane into a net called the *Voronoi diagram.* The construction of the Voronoi diagram for a pentagon is shown in Fig. 2.

Let the vertices of a polygon be $x^1, x^2, ..., x^N$ where

$$x^i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

The points v^{ij} such that

$$v^{ij} = \frac{x^i + x^j}{2}$$

will then be the extremities of the Voronoi diagram.

Given a polygon its Voronoi diagram is constructed and the points v^{ij} obtained. For high complexity polygons it is usually computationally more efficient to use this procedure than to directly evaluate v^{ij} from the x^i and x^j. The symmetry conditions are then determined as explained below.

The symmetry conditions for a polygon depend on whether it has an odd or even number of vertices.

(i) If the number of vertices is odd, the axis of symmetry should pass through a vertex and the midpoint of two other vertices.

(ii) If the number of vertices is even, the axis of symmetry passes through two vertices or two midpoints.

For an odd polygon, an axis of symmetry to exist and pass through a point x^k, the required condition is that there exist a set of x^i and x^j ($i \neq k, j \neq k$) such that

$$<x^i - x^j, v^{ij} - x^k> = 0$$

such x^i and x^j will form pairs of points symmetric with respect to the axis through x^k.

If there is to be no axis of symmetry through x^k, then

$$<x^i - x^j, v^{ij} - x^k> \neq 0$$

for $i \neq k$, $j \neq k$. This is a necessary and sufficient condition to guarantee the nonexistence of any axis of symmetry through x^k. The same procedure is repeated for all vertices to verify whether the polygon has any axis of reflection symmetry.

For an even vertex polygon, two different kinds of axes of symmetry can exist.

i) Axis passing through two vertices.

ii) Axis passing through two midpoints of vertices.

i) Let x^a and x^b be the vertices to be checked. An axis of symmetry will pass through these vertices if there exists a set of x^i and x^j such that

$$<x^i - x^j, v^{ij} - x^a> = 0$$

and

$$<x^i - x^j, v^{ij} - x^b> = 0$$

for $i \neq$ a or b, $j \neq$ a or b. Such x^i and x^j points will form pairs which are symmetric with respect to the axis joining x^a and x^b. If the line joining x^a and x^b is not to be an axis of symmetry then

$$<x^i - x^j, v^{ij} - x^a> \neq 0$$

or

$$<x^i - x^j, v^{ij} - x^b> \neq 0$$

for $i \neq$ a or b, $j \neq$ a or b. This is a necessary and sufficient condition to guarantee the nonexistence of any axis of symmetry through x^a and x^b . This procedure is repeated for all vertices to verify whether the polygon has any axis of symmetry.

ii) Let v^{pq} and v^{rs} be the vertex midpoints to be checked. An axis of symmetry will pass through these if there exists a set of x^i and x^j such that

$$<x^i - x^j, v^{ij} - v^{pq}> = 0$$

and

$$<x^i - x^j, v^{ij} - v^{rs}> = 0$$

for i $\neq$ p, q, r or s and j $\neq$p, q, r or s. Such x^i and x^j will form pairs of points symmetric with respect to the axis joining v^{pq} and v^{rs}.

If the line joining v^{pq} and v^{rs} is not to be an axis of symmetry then

$$<x^i - x^j, v^{ij} - v^{pq}> \neq 0$$

or

$$<x^i - x^j, v^{ij} - v^{rs}> \neq 0$$

for i $\neq$ p, q, r or s and j $\neq$ p, q, r or s. This is a necessary and sufficient condition to guarantee the nonexistence of any axis of symmetry through v^{pq} and v^{rs}. This procedure is repeated for all combinations of vertices to check that no axis of symmetry exists.

b) Having verified that no axis of symmetry exists for a polygon, we need to further verify whether any axis of symmetry will exist under any affine transformation of the face.

The conditions derived above for no axis of symmetry to exist are of the general form

$$U^T V \neq 0$$

where U and V are two dimensional vectors. Under an affine transformation A, the condition becomes

$$U^T A^T A V \neq 0$$

whether this condition will be satisfied or not depends on the nature of U and V i.e. on the nature of the polygon and also on what kind of affine transformation it is subjected to i.e. on A. Thus a nonsymmetric triangle can be affine transformed to an equilateral or isosceles triangle which has axis of symmetry.

c) If the face has an axis of symmetry as verified in a) or b) then it is subjected to distortion which removes the axes of symmetry. It has been found that while for any particular distortion there always exists an affine transformation that would yield an axis of symmetry, if the polygon is subjected to two separate distortions which are antisymmetric with respect to each other, there exists no affine transformation which yields axes of symmetry for both cases. If these two distortions are referred to as D_1 and D_2, then the procedure consists of subjecting the polygon to D_1 and checking it according to a) and b) to see whether it has any axis of symmetry. If it does it is subjected to D_2 and by the above argument it will not have any axis of symmetry.

It has been found that polygons with three or four vertices can always be affine transformed to symmetric polygons. So a *minimum of five points* are needed to obtain a nonsymmetric polygon. A technique has been developed whereby it is not necessary for all the five points to physically lie on the polygon. If we have three points on the polygon, the other two points can be obtained as functions of coordinates of these three points. This is shown for a triangle in Fig.3, where P_1, P_2 and P_3 are points on the triangle (its vertices) and Q_1 and Q_2 are artificially created points. The five points should be positioned such that the polygon formed by them is the distortion D_1 or D_2 referred to above. Two such distortions are shown in Fig.3 b and c.

4.3) **Experimental Results:** Fig.4 shows certain objects that have been used to test the simulation of the MIAG algorithm. Fig.5 shows the output of the program for recognition and attitude determination of an object under two different orientations. Fig.6, 7 and 8 refer to certain physical objects that have been used to test the MIAG algorithm. These objects are a simple polyhedral structure, a space shuttle model and a space station model. These figures show these objects and their thinned wireframes.

5) CONCLUSION:

The MIAG algorithm has been extended for the attitude determination of general polyhedral objects. The algorithm has been tested under conditions simulating space conditions and the results are presented in this paper. Work is in progress to extend the algorithm to the general case of recognising and localising any general object.

REFERENCES

[1] C.W.Richard and H.Hemani, "Identification of Three-Dimensional Objects using Fourier Descriptors of the Boundary Curve". *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-2, Mar.1980, pp127-136

[2] S.A.Dudani, K.F.Breeding, and R.B.McGee, "Aircraft Identification by Moment Invariants", *IEEE Trans. on Computers*, vol C-26, no.1, October 1977, pp. 39-45

[3] L.T.Watson and L.G.Shapiro, "Identification of Space Curves from Two-Dimensional Perspective Views", *IEEE Trans. Pattern Anal. and Machine Intell.* vol. PAMI-4, Sep.1982, pp.469-475

[4] D.Marr and T.Poggio, "Cooperative computation of stereo disparity", *Science*, vol.194, pp.283-287, 1977

[5] D.Marr, Vision: *A Computational Investigation into the Human Representation and Processing of Visual Information.* W.H.Freeman and Co., 1982

[6] O.D.Faugeras, M.Hebert, E.Pauchon, J.Pouce, "Object Representation and Positioning from Range Data," *Robotics Research*, First Intrl. Symp., 1984

[7] B.K.P. Horn "Robot Vision", MIT Press, 1986

[8] K.A. Stevens et al., "Understanding Images at MIT: Representative Progress". *Proc. DARPA Image Understanding Workshop*, 1980, pp 15-19

[9] J.R.Kender, "Shape from Texture: A Computational Paradigm", *Proc. DARPA Image Understanding Workshop*, 1979, pp.134-138

[10] A.P.Witkin, "Shape from Contour", Phd dissertation, MIT, Cambridge MA, 1980

[11] H.G.Barrow and J.M.Tenenbaum, "Interpreting line drawings as 3D surfaces" Proc. 1st Annu. Nat. Conf. AI", 1980

[12] Bamieh B. and deFigueiredo R.J.P., "A General Moment Invariants / Attributed-Graph Method for Three-Dimensional Object Recognition from a Single Image", *IEEE Journal of Robotics and Automation*, March 1986, pp 31-41

[13] Pavlidis T., "Algorithms for Graphics and Image Processing", Computer Science Press, 1982.

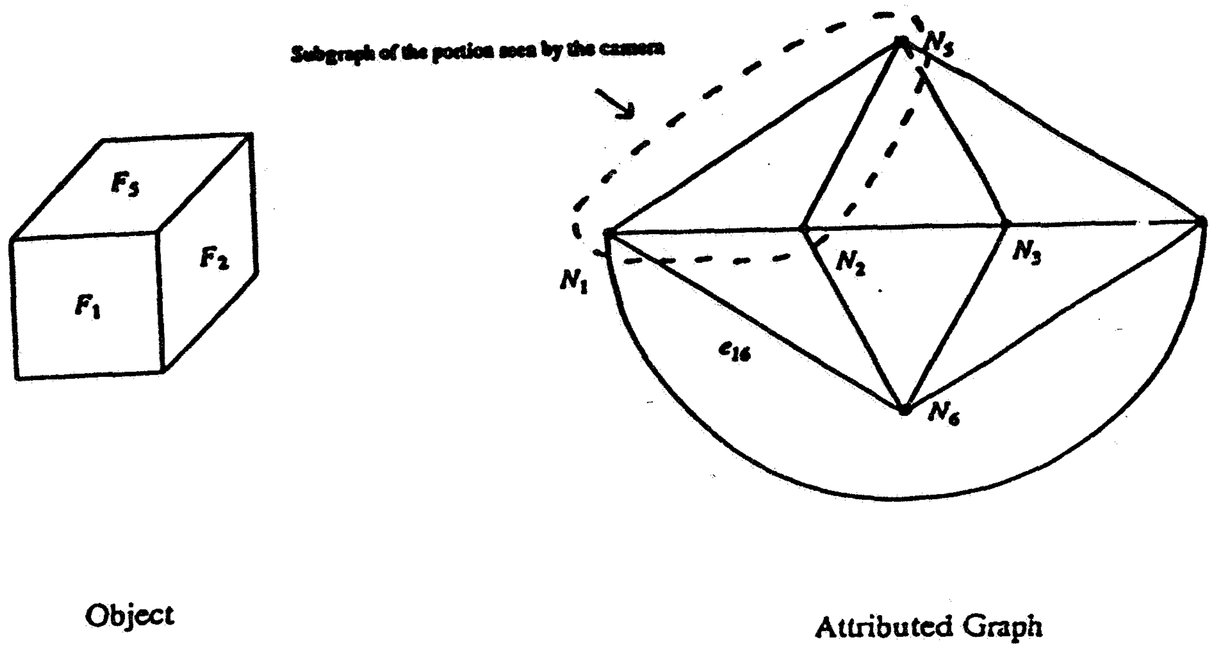


Fig.1 Wireframe of a cube and its attributed graph representation

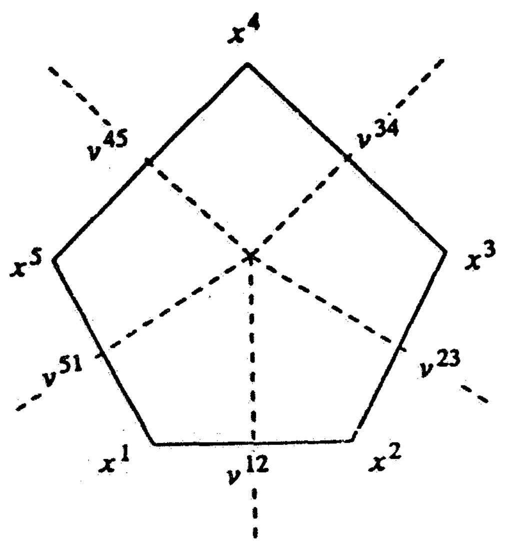


Fig.2 Voronoi diagram of a polygon

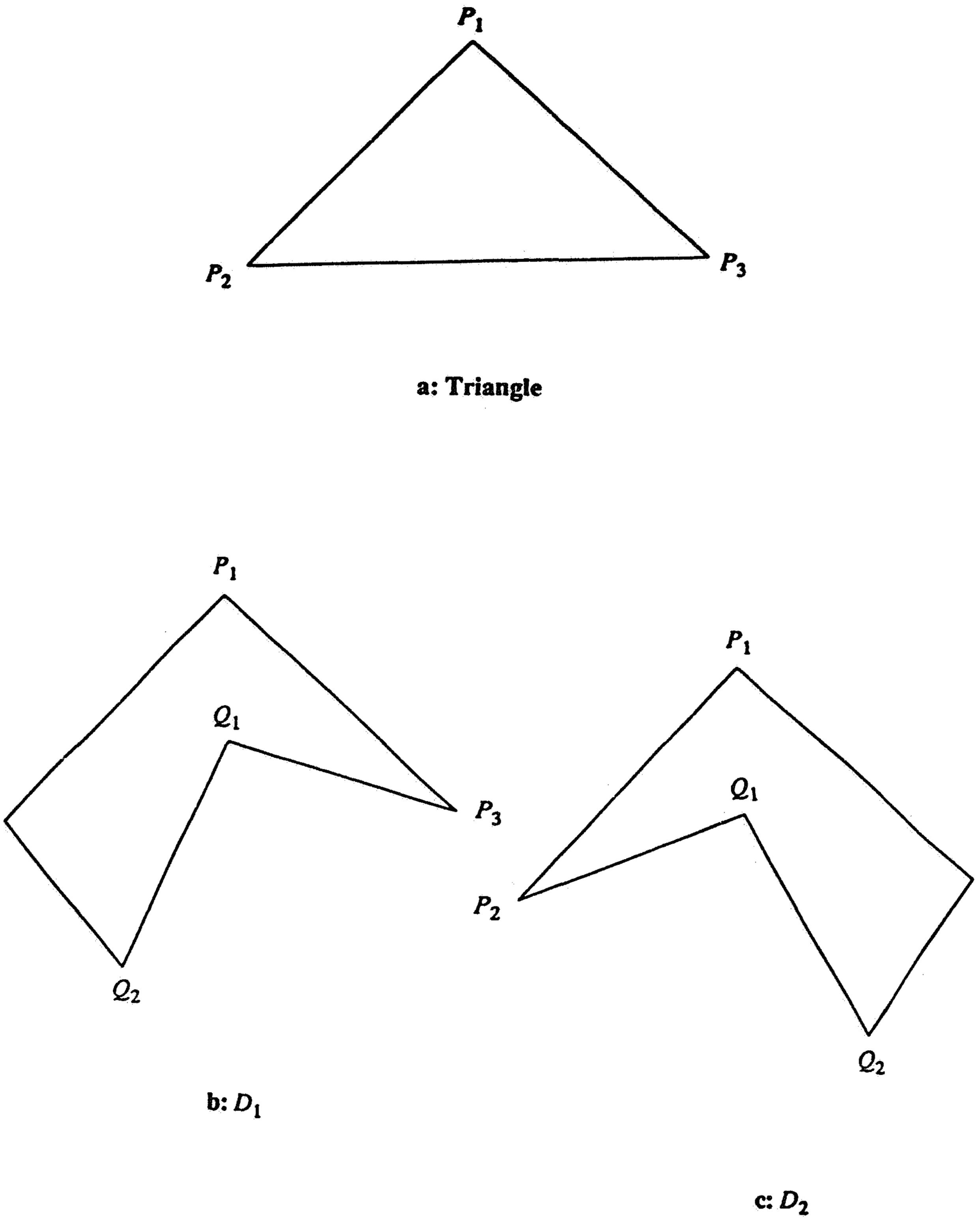


Fig 3: A triangle subjected to D_1 and D_2

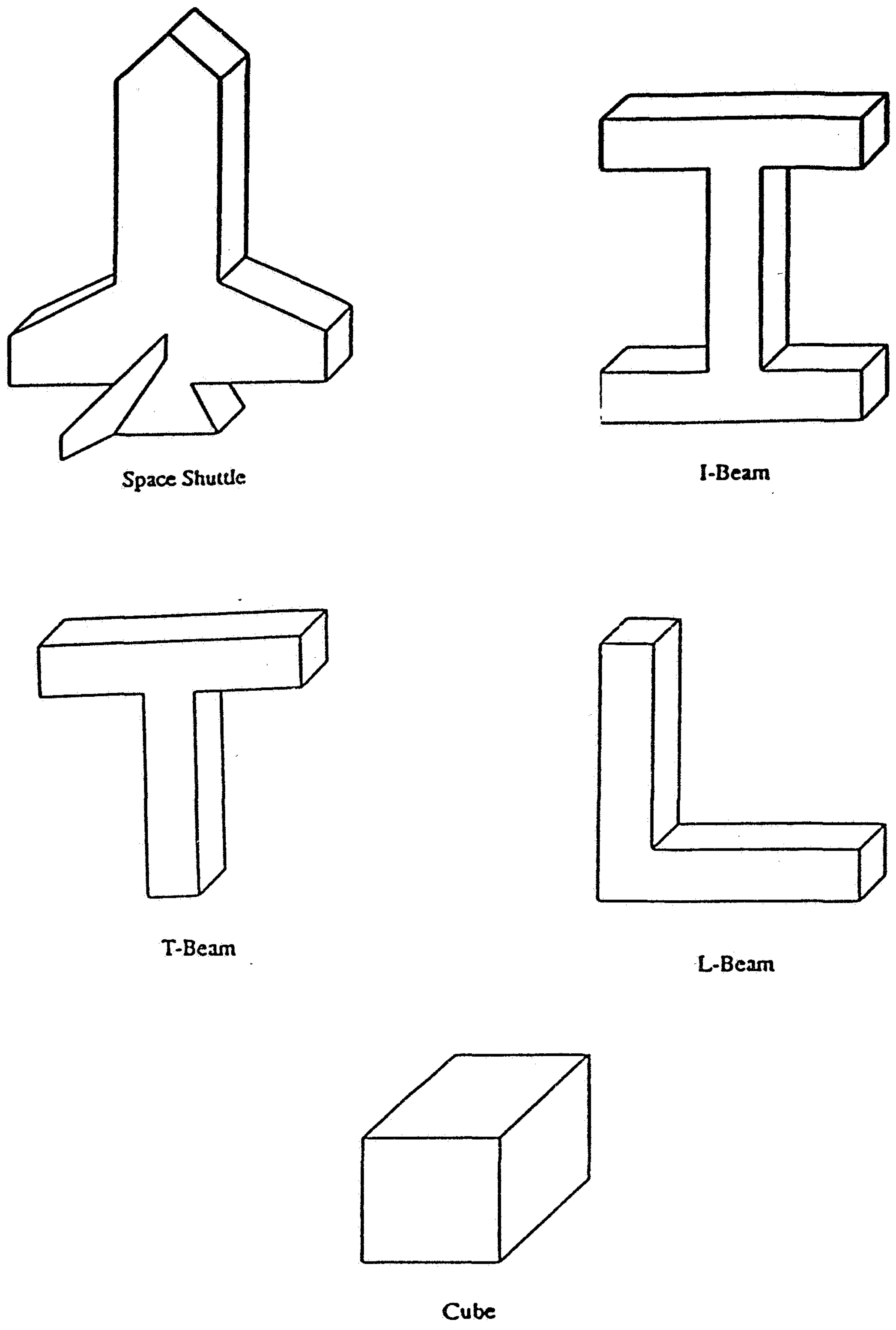
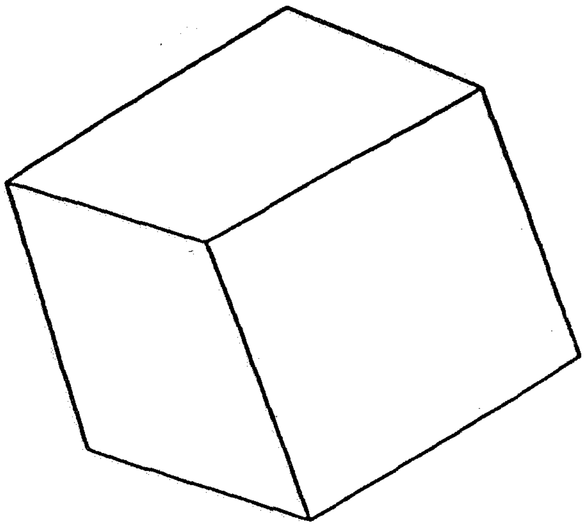


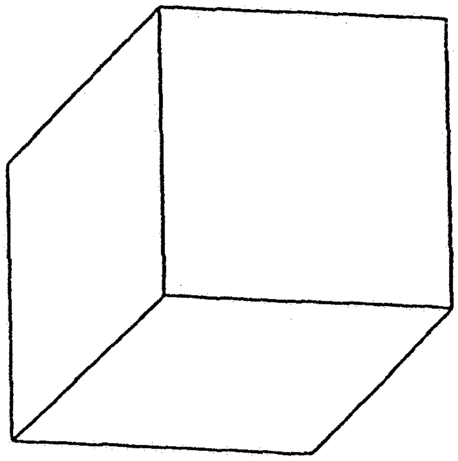
Fig.4 Examples of objects used to test the MIAG algorithm



```
roll = 45
pitch = 45
yaw = 45

Matched to cube

Match correspondences are:
Wireframe face # ---> Model face #
0 ---> 0
1 ---> 1
2 ---> 4
```



```
roll = 90
pitch = 30
yaw = 90

Matched to cube

Match correspondences are:
Wireframe face # ---> Model face #
0 ---> 0
1 ---> 1
2 ---> 5
```

Fig.5 Examples of Recognition and Attitude Determination using MIAG Algorithm

Object ---> Edge Detector Output ---> Edge Thinning Output

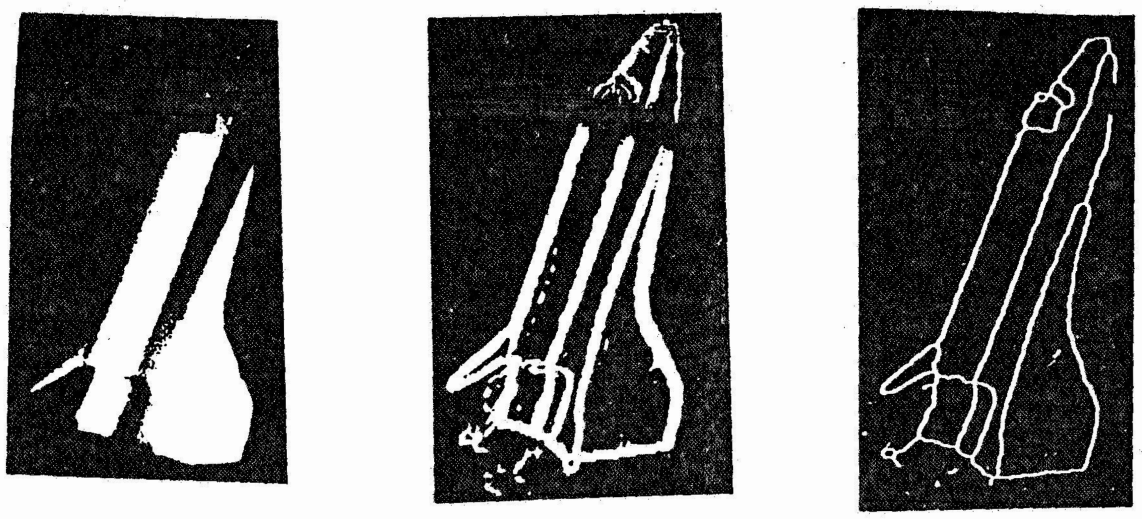


Fig.6 Space Shuttle

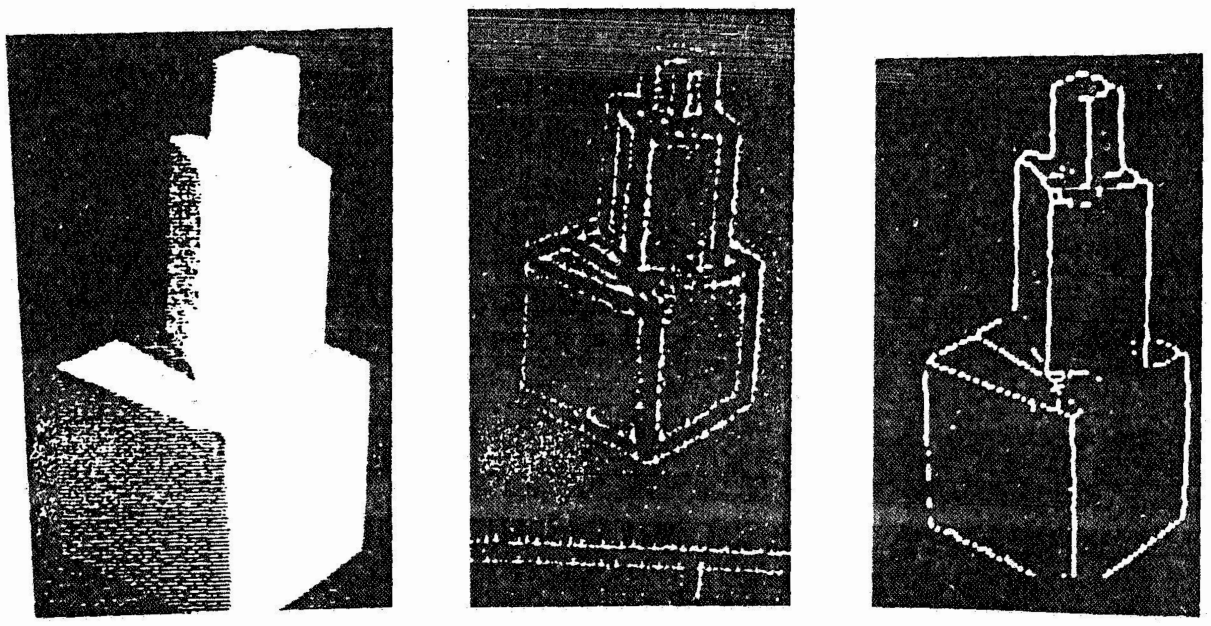
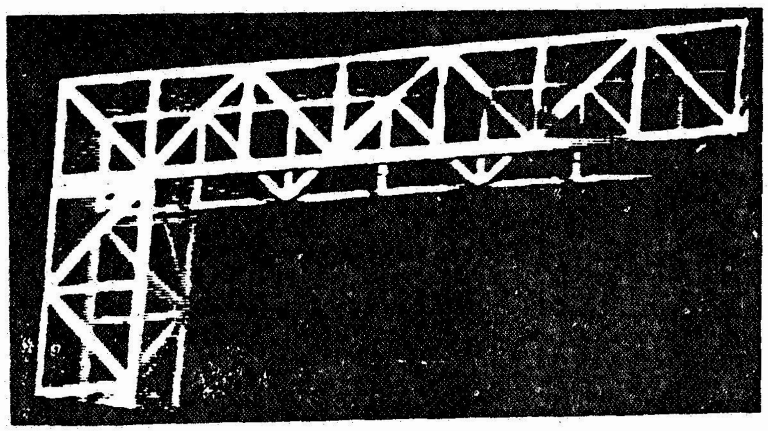
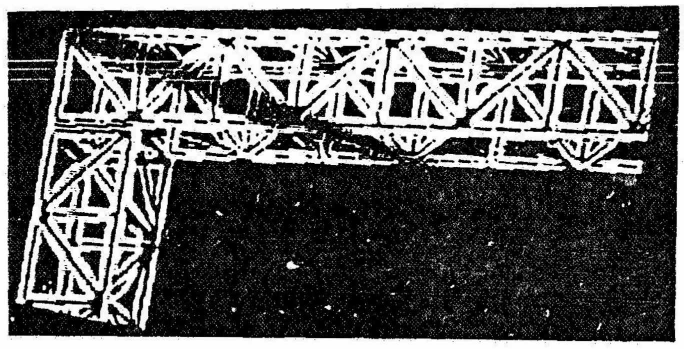


Fig.7 Polyhedral Structure

Object ---> Edge Detector Output ---> Edge Thinning Output





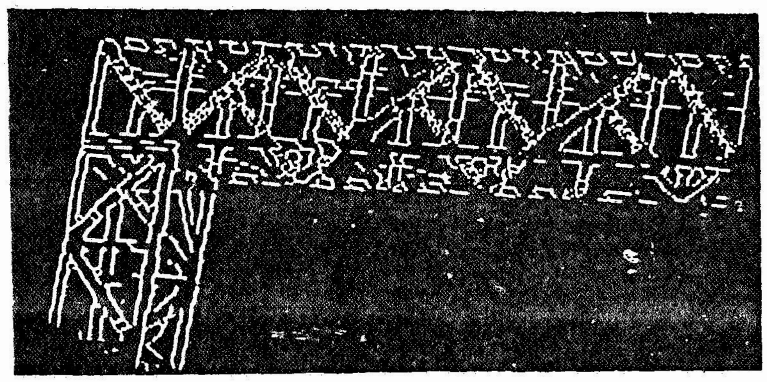


Fig.8 Part of Space Station

S15-63
60
188135

Differential Surface Models for Tactile Perception of Shape and On-Line Tracking of Features

H. Hemami
Ohio State University
Columbus, OH 43210

ØM 593208

1. Abstract

Tactile perception of shape involves an on-line controller and a shape perceptor. The purpose of the on-line controller is to maintain gliding or rolling contact with the surface, and collect information, or track specific features of the surface such as edges of a certain sharpness. The shape perceptor uses the information to perceive, estimate the parameters of, or recognize the shape. The differential surface model depends on the information collected and on the a-priori information known about the robot and its physical parameters. These differential models are certain functionals that are projections of the dynamics of the robot onto the surface gradient or onto the tangent plane. They involve the states of the robot (i.e., angles and angular velocities), input torques or forces to the robot, the coefficient of friction μ, and some of the differential properties of the surface such as the units of tangent and normal to the surface, gradient, Hessian, and the radius of curvature and its projections onto planes. A number of these differential properties may be directly measured from present day tactile sensors. Others may have to be indirectly computed from measurements. Others may constitute design objectives for distributed tactile sensors of the future. A parameterization of the surface leads to linear and nonlinear sequential parameter estimation techniques for identification of the surface. Many interesting compromises between measurement and computation are possible.

2. Introduction

Tactile perception of shape by natural systems has been the subject of many recent studies [1]. Tactile perception in robotic systems requires maintenance of gliding and/or rolling contact with the unknown object and infering information about the shape. A major component of this kind of probing is the controller. The controller needs on-line construction of the kinematics [2], force feedback [3], and inverse dynamics [4] to generate the needed input torques to the robot joints. The available tactile sensors to date, however, are not adequate for fast and efficient execution of rolling and gliding manipulations [5,6]. Once gliding or rolling is maintained, the perception of shapes involves using kinematic, and dynamic information to gather information about the manipulated object [7]. The process of determining the shape involves availability of a-priori computational and symbolic models of shape [8].

For smooth surfaces, that are linear in an unknown parameter vector, linear sequential estimation algorithms can be used to arrive at these parameters [9,10,11]. Alternatively, solution of partial differential equations or nonlinear estimation algorithms are needed [10].

When the object or surface is known, the trajectory of the robot end effector can be a-priori determined the control of both gliding [12,13] and rolling [14] on known surfaces has been studied before. Brown [15] has considered the control problem for gliding on unknown objects. This paper deals with the kinematics and dynamics of gliding and rolling contact of a known end effector gliding and/or rolling on an unknown surface. Two differential surface models for perception are derived. For parametric surfaces a linear sequential estimation algorithm is sketched.

3. The Kinematic Problem

The on-line kinematic problem for purposes of gliding and rolling on an unknown surface is discussed here by a simple two rigid body problem. A planar rigid body end effector is considered that maintains contact with an unknown rigid body by gliding or rolling on it (Fig. 1). The internal coordinate system of the end effector is the y_1y_2 axes centered at the center of gravity of the end effector A and parallel with the principal axes of the end effector. The smooth surface of the end effector is assumed to be known implicitly or parametrically in its own coordinate system.

$$C(Y) = 0 \qquad (1.a)$$

$$Y = Y(\alpha) = [y_1(\alpha),\ y_2(\alpha)]^{\tau} \qquad (1.b)$$

and the point of contact B (in gliding) is specified by α_1:

$$Y_B = Y(\alpha_1) \tag{2}$$

Similarly the smooth unknown surface may be characterized parameterically or implicity. An implicit representation is assumed here:

$$D(X) = D(x_1, x_2) = 0 \tag{3}$$

Let the coordinates of A be given by the two-vector X_A. The coordinates of the contact point B in the inertial coordinate system are

$$X_B = X_A + \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 \\ \sin\theta_3 & \cos\theta_3 \end{bmatrix} Y_B \tag{4}$$

The control requirements for gliding contact are:

1. existence of the normal contact force to making sure that the contact is maintained,
2. knowledge about the point of contact,
3. guiding the motion of the end effector, along the unknown surface,

and finally, 4. knowledge of the radius of curvature of the unknown surface.

along the unknown surface. A control input to this guidance is the tangential velocity of contact $v_B(t)$. The latter guidance requires sensing of the unit tangent vector T at B in the end effector coordinate system and transforming it to the inertial coordinate system

$$\dot{X}_B(t) = v(t)T \tag{5}$$

Differentiating Eq. (4) with respect to time and substituting in Eq. (5) gives

$$\dot{X}_A(t) = v(t)T + \dot{\theta}_3 \begin{bmatrix} \sin\theta_3 & \cos\theta_3 \\ -\cos\theta_3 & +\sin\theta_3 \end{bmatrix} Y_B \tag{6}$$

Eq. (6) relates the local translational velocities $\dot{X}_A$ to the angular velocity of the end effector $\dot{\theta}_3$.

Another interpretation of Eq. (6) is that the terms on the right side of Eq. (6) are respectively a small translation of point A and a small rotation of point A about point B. The angular velocity $\dot{\theta}_3$ itself is a function of the local curvature of the unknown surface at the point of contact. Let ds be the traversed distance on the unknown surface. By definition, the radius of curvature is given by

$$\rho_u = \frac{1}{d\theta_3/ds} = \frac{ds}{d\theta_3} \tag{7}$$

$$\dot{\theta}_3 = \frac{d\theta_3}{dt} = \frac{1}{\rho_u}\frac{ds}{dt} = \frac{v(t)}{\rho_u} \tag{8}$$

Therefore, Eqs (6) and (8) together define the instantaneous kinematics of the gliding motion.

In the rolling motion the contact point moves on the end effector as well as on the unknown surface so that the incremental distances traversed on both surfaces are equal. In addition to the four requirements of gliding motion as before, the end effector should have knowledge of its own local radius of curvature at the point of contact. Assume $v(t) = ds/dt$ is the specified control input, and assume a convex surface, and a convex end effector surface as in Fig. 1. It is not difficult to show that

$$\dot{\theta}_3 = v(t)\left(\frac{1}{\rho_e} + \frac{1}{\rho_u}\right) \tag{9}$$

where ρ_e and ρ_u are respectively the local radii of curvature of the end effector and the unknown surface at the point of contact.

Similarly from the incremental form of Eq. (4) and the definition of rolling, it follows that

$$dX_A = \begin{bmatrix} +\sin\theta_3 & \cos\theta_3 \\ -\cos\theta_3 & \sin\theta_3 \end{bmatrix} d\theta_3\, Y_B \quad (10) \qquad \text{or} \qquad \dot{X}_A = \dot{\theta}_3 \begin{bmatrix} \sin\theta_3 & \cos\theta_3 \\ -\cos\theta_3 & \sin\theta_3 \end{bmatrix} Y_B \quad (11)$$

To summarize, Eqs (9) and (11) are the instantaneous kinematics of the rolling motion.

If the unknown surface is convex, Eq. (9) is replaced by

$$\dot{\theta}_3 = v(t)\left(\frac{1}{\rho_e} - \frac{1}{\rho_u}\right) \tag{12}$$

Suppose the center of gravity of the end effector is connected to a two link robot (Fig. 2).

$$X_A = \begin{bmatrix} \ell_1\cos\theta_1 + \ell_2\cos\theta_2 \\ \ell_1\sin\theta_1 + \ell_2\sin\theta_2 \end{bmatrix} \tag{13}$$

Differentiating Eq. (13) with respect to time gives

$$\dot{X}_A = \begin{bmatrix} -\ell_1\sin\theta_1 & -\ell_2\sin\theta_2 \\ \ell_1\cos\theta_1 & + \ell_2\cos\theta_2 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} \tag{14}$$

The latter equation provides the instantaneous kinematic of the three-link system.

$$\dot{\theta} = [\dot{\theta}_1 \ \dot{\theta}_2 \ \dot{\theta}_3]^\tau \tag{15}$$

for either of the gliding or rolling motion.

From the above discussion it can be stated that measuring or estimating the local radius of curvature ρ_u of the unknown object and determining convexity or concavity are two important parameters for the kinematics of rolling or gliding. The local radius of curvature has two more uses. 1) It is needed for an ideal inverse dynamics systems where the accelerations are needed to construct the input torques [11]. It is needed for detecting snarp edges (small ρ_u) and consequent tracking of such sharp edges on three dimensional surfaces.

4. The Dynamics Problem for Point Contact

Consider the three-link planar robot of Fig. 2 with no contact with any object or surface, the equations of motion for this system [12,13] are

$$J(\theta)\ddot{\theta} + B(\theta)\dot{\theta}^2 + E(\theta) = CU \tag{16}$$

where U is the vector of torque actuators at the joints. Suppose the gliding is on a frictionless surface. The contact force is along the unit normal vector N to the surface, and assume its magnitude is γ. The incremental motion of the contact point on the robot is

$$dX_B = dX_A + \begin{bmatrix} -\sin\theta_3 & -\cos\theta_3 \\ \cos\theta_3 & -\sin\theta_3 \end{bmatrix} Y_B \, d\theta_3$$

Let N be resolved in the inertial coordinate system. The incremental work of the contact force is

$$dW = < \gamma N, dX_B >$$

where < > is the inner product, and

$$\frac{dW}{d\theta} = < \gamma N, \frac{dX_B}{d\theta} >$$

where

$$\frac{dX^\tau_B}{d\theta} = \begin{bmatrix} -\ell_1\sin\theta_1 & \ell_1\cos\theta_1 \\ -\ell_2\sin\theta_2 & \ell_2\cos\theta_2 \\ -y_{1B}\sin\theta_3 - y_{2B}\cos\theta_3 & +y_{1B}\cos\theta_3 - y_{2B}\sin\theta_3 \end{bmatrix}$$

The equation of motion with the contact in effect are:

$$J(\theta)\ddot{\theta} + B(\theta)\dot{\theta}^2 + E(\theta) = CU + \begin{bmatrix} \frac{dX^\tau_B}{d\theta} \end{bmatrix} \gamma N \tag{17}$$

The holonomic constraint governing the dynamics is

$$D(X_R) = 0 \tag{18}$$

Differentiating Eq. (18) with respect to time gives

$$\dot\theta^\tau \frac{dX_R{}^\tau}{d\theta}\frac{dD}{dX} = \frac{dD^\tau}{dX}\frac{dX_R}{d\theta}\dot\theta = 0 \tag{19}$$

A final relation that is important for the analysis to follow is the definition of the unit normal vector N. The gradient vector of the unknown surface is dD/dX and by definition

$$N = \frac{dD}{dX} \Big/ \left\| \frac{dD}{dX} \right\| \tag{20}$$

where $\| \; \|$ is the Euclidean norm.

Consider the rolling type of constrained motion. Let the magnitude of the tangential constraint force be λ. The contribution of these forces to the equations of motions is

$$\frac{dW}{d\theta} = \left< \frac{dX_R}{d\theta}, (N\gamma + T\lambda) \right> \tag{21}$$

For the rolling motion, there are two constraints. The holonomic contact constraint of Eq. (18) and the non-holonomic roll constraint --no motion along T at the point of contact.

$$T^\tau \frac{dX_R}{dt} = T^\tau \frac{dX_R}{d\theta}\dot\theta = 0 \tag{22}$$

Eq. (19) implies no motion of the contact point along the unit normal vector. Eq. (22) means no motion of the contact point along the unit tangent vector. Consequently the only possible motion is a rotation about the contact point and hence a rolling motion.

In order for rolling to occur and no slippage or gliding to take place, the coefficient of friction μ must be different than zero and the forces of constraint must be governed by

$$0 < \lambda < \mu\gamma \tag{23}$$

5. Differential Surface Models

In this section constituent relations between the state θ, $\dot\theta$, the input U, the forces and the surface geometry are derived. If the surface is known, these equations can be used to solve for the forces of constraint γ and λ. If alternatively, the forces are known, these equations can be used as differential surface models, and used for estimating the shape of the surface. These constituent relations are arrived at by differentiating the constraint Eqs. (18) and (22) with respect to time and eliminating the acceleration $\ddot\theta$ between the latter second derivatives and the equations of motion.

The above procedure could be carried out simultaneously for both constraints. However, it is done for the individual constraints here in order to demonstrate two alternative formulations, one more analytical, one slightly more suitable for computational purposes.

5.1 First Formulation.

The differentiation of Eq. (18) with respect to time gives:

$$\dot\theta^\tau \frac{dX_R{}^\tau}{d\theta}\frac{dD}{dX} = 0 \tag{24}$$

This equation implies no velocity component exists along the unit normal. An alternative form for Eq. (24) is

$$\dot X_R{}^\tau(t)N = 0 \tag{25}$$

The differentiation of Eq. (24) with respect to time gives

$$\ddot\theta^\tau \frac{dX_R{}^\tau}{d\theta}\frac{dD}{dX} + \dot\theta^\tau \frac{dX_R{}^\tau}{d\theta}\frac{d^2D}{dX^2}\frac{dX_R}{d\theta}\dot\theta + \dot\theta^\tau \frac{\partial}{\partial\theta}\left[\frac{dD^\tau}{dX}\frac{dX_R}{d\theta}\right]\dot\theta = 0 \tag{26}$$

Elimination of $\ddot\theta$ between Eqs. (26) and the dynamics of the system

$$J\ddot\theta + B(\theta)\dot\theta^2 + E(\theta) = CU + \left< \frac{dX_R}{d\theta}, N\gamma + T\lambda \right> \tag{27}$$

gives the first form of the constituent equation

$$-\dot{\theta}^{\tau}\frac{dX_{B}^{\tau}}{d\theta}\frac{d^{2}D}{dX^{2}}\frac{dX_{B}}{d\theta}\dot{\theta} - \dot{\theta}^{\tau}\frac{\partial}{\partial\theta}\left[\frac{dD^{\tau}}{dX}\frac{dX_{B}}{d\theta}\right]\dot{\theta} = \frac{dD^{\tau}}{dX}\frac{dX_{B}}{d\theta}J^{-1}\left[-B\dot{\theta}^{2} - E\theta + CU + \frac{dX^{\tau}_{B}}{d\theta}(N_{\gamma} + T\lambda)\right] \quad (28)$$

If all parameters and quantities in Eq. (28) are available, it is a constituent relation in the unknown gradient and Hessian dD/dX and $d^{2}D/dX^{2}$. If N and T are also not available, it is easy to include Eq. (20) for N and the following for T

$$T = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} N \quad (29)$$

The result is a more complex constituent relation. Because Eq. (28) involves the first and second partial derivatives of the surface, it is a differential model of the unknown surface. If λ and γ are negligibly small, and/or if $\dot{\theta}$ --the angular velocities --are relatively small, Eq. (28) simplifies.

5.2 Second Formulation. Eq. (28) establishes one relation among γ and λ. A second relation among λ and γ could be obtained in a similar fashion from the second constraint equation. However, an alternative formulation is given here.

If Eq. (22) is differentiated with respect to time, one obtains

$$T^{\tau}\frac{dX_{B}}{d\theta}\ddot{\theta} + \dot{\theta}^{\tau}\frac{d}{d\theta}\left(T^{\tau}\frac{dX_{B}}{d\theta}\right)\dot{\theta} = 0 \quad (30)$$

Between Eqs. (27) and (30) one can eliminate $\ddot{\theta}$ to obtain the second differential surface model

$$-\dot{\theta}^{\tau}\frac{d}{d\theta}\left(T^{\tau}\frac{dX_{B}}{d\theta}\right)\dot{\theta} + T^{\tau}\frac{dX_{B}}{d\theta}J^{-1}\left(B\dot{\theta}^{2} + E\right) = T^{\tau}\frac{dX_{B}}{d\theta}J^{-1}\left[CU + \frac{dX^{\tau}_{B}}{d\theta}(N_{\gamma} + \lambda T)\right] \quad (31)$$

The above two differential surface models, are two independent Eqs. in γ and λ. If everything else is known including the unknown surface, these equations can be solved for these constraint forces as functions of the state $[\theta^{\tau}, \dot{\theta}^{\tau}]^{\tau}$, input U and the surface $D(X) = 0$. They provide differential information about the surface, if everything else including γ and λ is known.

6. Shape Perception by Parameter Estimation

Suppose the unknown surface is representable by a vector of unknown parameters β.

$$D(X) = P^{\tau}(X)\beta = 1 \quad (32)$$

one may use coordinates of the many contact point X_B to arrive at a system of linear equations in β, under the above assumption, the gradient and the Hessian are also linear in β. As a result the constituent differential surface models, sampled at some interval T of time provide independent information about vector β. These systems of over specified linear equations can be solved for β. Let the overspecified system be

$$M\beta = N \quad (33)$$

where each row of the equation is one additional piece of information from sampling the constituent surface models or Eq. (32), etc. (an example is worked out in [10]). The best estimate for β is the mean square error sense [9] is

$$\beta = (M^{\tau}M)^{-1}M^{\tau}N \quad (34)$$

This equation is also robust with respect to a certain amount of independent random measurement noise.

7. Acknowledgment

This work was supported by the Department of Electrical Engineering, The Ohio State University, Columbus, Ohio 43210. The author is indebted to Professor H. C. Ko, Chairman of the Department of Electrical Engineering, The Ohio State University for his sustained support and encouragement of this work.

8. References

[1] Gordon, G., Editor, Active Touch: The Mechanism of Recognition of Objects by Manipulation, Pergamon Press, Oxford, 1978.

[2] Whitney, D.E., "Resolved Motion Rate Control of Manipulator and Human Prostheses," IEEE Transactions on Man-Machine Systems, vol. 10, no. 2, 1969, pp. 47-53.

[3] Raibert, M.H. and Craig, J.J., "Hybrid Position/Force Control of Manipulators," Journal of Dynamic Systems, Measurement, and Control, vol. 102, 1981, pp. 126--133.

[4] Paul, R.P., Shimano, B., and Mayer, G.E., "Differential Kinematic Control Equations for Simple Manipulators," IEEE Transactions on Systems, Measurement, and Control, vol. 11, no. 6, 1981, pp. 456-460.

[5] Hackwood,S and Beni, G., "Sensor and High Precision Robotics Research," Proceedings of the 1st Intl. Symposium on Robotic Research, pp. 529-545, Cambridge, MA 1984, edited by M. Brady and R. Paul, MIT Press.

[6] Harmon, L.D., "Automated Tactile Sensing," International Journal of Robotics Research, vol. 1, no. 2, pp. 3-32, 1982.

[7] Bajcsy, R., "What Can We Learn from One Finger Experiments," The 1st Intl. Symposium on Robotic Research, pp. 509-527, Cambridge, MA, 1984, edited by M. Brady and R. Paul, MIT Press.

[8] Faugeras, O. and Hebert, M., "The Representation, Recognition, and Locating of 3-D Objects," International Journal of Control, vol. 5, no. 3, pp. 27-52, 1986.

[9] Lee, R.C.K., Optimal Estimation, Identification, and Control, MIT Press, Research Monograph, no. 28, 1964.

[10] Hemami, H. and Goddard, R.E., "Recognition of Geometric Shape by a Robot System," 1987, to appear in Journal of Robotic Systems.

[11] Hemami, H., "Shape Determinism by Tactile Sensing," Proc. of the National Science Foundation Workshop on Machine Dynamics, Denver, CO, Aug. 10-13, 1986, edited by A.H. Soni.

[12] Han, J.-Y., Stability, Contact Force and Adaptive Trajectory Control of Natural and Robotic Systems, Ph.D. Dissertation, The Ohio State University, March 1986.

[13] Buchner, H.J., Hines, M.J., and Hemami, H., "A Mechanism for Touch Control of a Sagittal Five-Link Hand-Finger," IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-15, no. 1, pp. 69-77, 1985.

[14] Wongchaisuwat, C., Hemami, H., and Buchner, H.J., "Control of Sliding and Rolling at Natural Joints," Journal of Biomedical Engineering, vol. 106, pp. 368-375, 1984, ASME Transactions.

[15] Brown, D.B., "On-Line Exploration of an Unknown Surface by a Three-Link Planar Robot," M.S. Thesis, The Ohio State University, June 1986, Report No. TH-86-HH-142.

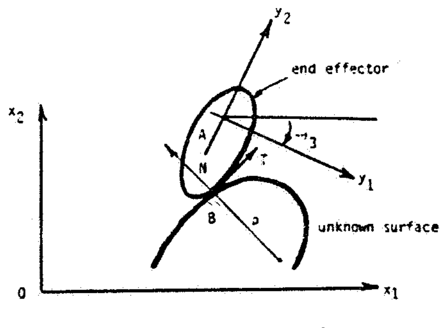


Figure 1: The planar two-body contact problem.

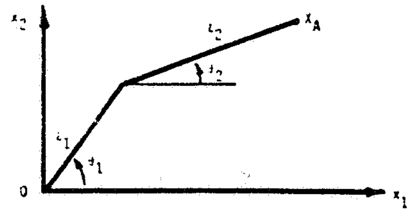


Figure 2: Gross motion of the end effector center of gravity by a two-link robot.

516-63
48
188136

Constraint-Based Stereo Matching

D.T. Kuan
FMC Corporation
Santa Clara, CA 95052

F4 563409

ABSTRACT

The major difficulty in stereo vision is the correspondence problem that requires matching features in two stereo images. ~~In this paper,~~ We describe a constraint-based stereo matching technique using local geometric constraints among edge segments to limit the search space and to resolve matching ambiguity. Edge segments are used as image features for stereo matching. Epipolar constraint and individual edge properties are used to determine possible initial matches between edge segments in a stereo image pair. Local edge geometric attributes such as continuity, junction structure, and edge neighborhood relations are used as constraints to guide the stereo matching process. The result is a locally consistent set of edge segment correspondences between stereo images. These locally consistent matches are used to generate higher-level hypotheses on extended edge segments and junctions to form more global contexts to achieve global consistency.

INTRODUCTION

Stereo vision is the process of reconstructing 3-D depth information from 2-D images. Depth information is crucial in passive navigation and scene interpretation applications. The key problem in stereo vision is to establish a correspondence between features in two images in order to calculate position in 3-D space according to stereo imaging geometry. A scene point will project to two image planes through cameras. The epipolar lines are intersections of the two image planes with an epipolar plane defined by the scene point and the two camera foci. Based on this relationship, if we locate one feature point in one image, then the corresponding feature in the other image must lie on the corresponding epipolar line. The displacement between two corresponding feature points is termed disparity.

There are basically two types of automated stereo matching techniques - area-based and feature-based. Early work on stereo vision used area-based cross-correlation techniques for image correspondence. For example, Moravec [5] used "interest points" and image intensity cross-correlation measure for stereo matching. An interest point is an image feature with significant intensity variation around it and is usually a corner point of an object. A set of interest points are extracted from one image and the corresponding features in the other image are searched using a hierarchical correlation technique. No global consistency is checked in the matching process. For images with similar intensity variation and no significant occlusion effect, the area-based stereo analysis technique works well. However, the technique may fail in the presence of repetitive features, surface discontinuity, and intensity variation.

The edge-based stereo technique first extracts edge features from both stereo images and uses various constraints to resolve the correspondence problem. Grimson [3] implemented an edge-based stereo algorithm with a coarse-to-fine strategy and the uniqueness and continuity constraints originally proposed by Marr and Poggio [4]. The zero-crossing edge pixels are used as features, and disparity similarity is used to determine the final matches. Recent work by Baker and Arnold [1,2] incorporates geometric constraints into the dynamic programming algorithm to match edge pixels in a single scanline and uses an edge connectivity constraint to guide the inter-scanline matching. Due to the limitation of the dynamic programming algorithm, edge pixel correspondence between the left and right images has a strict order sequence and edge reversal is not allowed. The advantages of edge-based stereo include faster processing speed (because it requires fewer features to match) more accurate results (because edges may be located with sub-pixel precision), and less sensitivity to intensity variation (because edges represent geometric features).

In this paper, we use the edge segment instead of the edge pixel as the primitive image feature. This choice has several advantages. The edge segment is a group of consistent edge pixels and is a more stable and robust feature primitive. The inter-scanline continuity constraint used in [2] is implicitly imbedded in the edge segment primitive. Vertical disparity problem that exists in edge-pixel based stereo algorithms does not happen here. In addition, the number of edge segments is much less than the number of edge pixels. This significantly reduces the computation time for stereo matching. Finally, geometric relations among edge segments can be used as constraint in stereo matching.

In our approach, edge pixels are first detected and linked into edge segments. Edge segment orientation and intensity profile are used as matching properties. Junctions are detected and classified according to their types. A junction is a place to propagate geometric constraints from one edge to all the connecting edges. The edge neighborhood relation is also used as a local geometric feature to propagate constraints among edges where a junction feature does not exist.

In the initial matching stage, standard epipolar constraint and individual edge properties are used to determine possible initial matches between edge segments in the left and right images. For each initial hypothesis, we then apply a set of geometric constraints to reduce matching ambiguity. The results of constraint checking are recorded and a maximum likelihood scheme is used to select the most likely match for each edge segment in stereo images. These locally consistent matches then generate higher-level hypotheses on extended edges and junctions to form more global contexts. The number of higher-level hypothesis is very small compared to the number of initial hypothesis because local matches have constrained the possible global matches. If a higher-level hypothesis has enough supports from local matches, it becomes a global match and can enforce global consistency to correct inconsistent local matches in the same context.

IMAGE FEATURE EXTRACTION

The image feature extraction process proceeds concurrently on two stereo images. Edge segments are first extracted, and then junctions and edge neighborhood relations are extracted based on edge segment relations.

The Edge Feature

The two stereo images are first smoothed by using a Gaussian convolution mask to reduce image noise. Edges are detected by using an eight-directional fast compass edge detector. Edge direction and magnitude information are used to thin and link edge pixels into edge chains. Then a recursive line-fitting algorithm is used to represent each edge chain as a sequence of line segments. Each sequence of line segments is stored in an extended edge structure and each line segment is represented in an edge segment structure. For each edge segment, its starting and ending points, mid-point, orientation, length, and average intensities on two sides of the edge are stored in the edge segment structure. Edge segments are indexed by dividing an image into a set of square windows to provide fast access to neighboring edges. This spatial indexing scheme speeds up local geometric feature calculation significantly. Figure 1 shows a pair of stereo images. Figure 2 shows all edge segments detected in Figure 1.

The Junction Feature

Junctions are detected by searching for all edge segments intersecting a small window attached at the ends of an edge segment. Junction type, orientation, location, associated edges, and relative edge angles are stored in the junction structure. Junction type includes L, arrow, fork, T, and a complex junction containing more than three edges. Edges associated with an junction are ordered according their orientations. A junction is a place to propagate constraints from one edge to all the connecting edges.

The Edge Neighborhood Feature

Junctions are useful features to propagate constraints between edges. However, most images contain few junctions. We use the neighborhood relation among edges as another class of local features to propagate constraints among edges. Each edge has a set of left-neighboring edges and a set of right-neighboring edges. A neighboring edge is defined as an edge that has significant vertical overlap with and is adjacent to a specified edge. The relative orientation, interval between an edge and its neighboring edges, and their vertical overlap interval, are stored in the edge neighborhood structure as a local feature for stereo matching.

CONSTRAINTS FOR STEREO MATCHING

In principle, each edge segment in the left image can match any edge segment in the right image. In practice, edge segment properties, stereo imaging geometry, edge continuity, and local geometric relations greatly constrain the possible matches for each edge segment. We describe in this section the constraints used in our stereo matching process.

The Epipolar Constraint

The standard epipolar constraint specifies that if an image feature exists in the left image, then the corresponding feature in the right image must lie on the epipolar line of the right image. Without loss of generality, we assume the two stereo images are properly aligned such that the epipolar lines are the scan lines of the image. In this case, for each edge in the left image, we only need to consider those right image edges having vertical overlap with the left image edge. The epipolar constraint is a strong geometric constraint based on stereo imaging geometry and has to be satisfied by all matches.

The Disparity Range Constraint

For each edge segment in the left image, we only search candidate edges within the window defined by the maximum allowed disparity interval in the right image. This constraint is applied in the initial matching stage.

The Edge Orientation Constraint

In general, the corresponding edge segments in stereo images should have similar orientations. The edge orientation constraint restricts each edge segment in the left image to match only those edges in the right image within an orientation threshold.

The Continuity Constraint

In the line-fitting procedure, it is possible that the corresponding edge of an edge segment is segmented into several pieces. To deal with this situation, partial edge segment matches must be considered. One edge segment may match more than one edge as long as these matches do not overlap in the vertical direction. In addition, these matches should have similar disparity to guarantee the corresponding edges are colinear. In the continuity constraint, if an initial match between two edge segments in the stereo images exists, and the two edges only partially overlap in the vertical direction, then we must find continuity evidence to support the partial match. We currently implement this by finding an edge connected to and colinear with the partially overlapped edge in the direction of the required extension. This constraint also applies to the complete edge segment match case where two edge segments have significant vertical overlap. In this case, only continuity in the correct direction is checked.

The Disparity Compatibility Constraint

Edges that are connected or close to each other in the image usually have similar disparities. This is based on the smoothness constraint of physical objects. In this constraint, we first identify all the edges that are connected or close to a given edge in one image by using junction structure and edge neighborhood relations. We then look for supporting matches from this edge group with similar disparity to one of the matches of the given edge, and record this information for global consistency checking.

The Junction Constraint

In the junction constraint, if two edges in the left image are in the same junction, we then check the corresponding matches in the right image to see if they are in the same junction. The relative edge segment ordering and angles in the junction are also checked to further reduce ambiguity.

The Neighborhood Relation Constraint

Each edge segment has its left and right neighbors. Unless there is occlusion or edge reversal effects, this edge neighborhood relation will be preserved in both stereo images. In the neighborhood relation constraint, if two edges are neighboring edges in the left image, then the corresponding matches in the right image should also be neighboring edges. The local features in the edge neighborhood structure are used for matching.

STEREO MATCHING

We separate the stereo matching process into three stages. In the initial matching stage, for each edge segment in the left image, we apply the epipolar constraint, disparity range constraint, and edge orientation constraint to obtain a set of possible matches. In Figure 2, 21 edge segments have no match in the right image, 38 edges have a unique match, 63 edges have two matches, 55 edges have three matches, and 153 edges have more than three matches. Most no-match edges are small edge segments or horizontal edge segments that are not very useful for stereo depth reconstruction. The constraints applied in this stage only use single edge segment properties to limit the search space.

In the second matching stage, we propagate constraints through junctions and neighboring edges to resolve multiple matching ambiguity of each edge segment. For each edge segment, a weighted average of the results after applying constraints and the edge segment similarity measure is used to calculate the likelihood of a hypothesized match. The most likely match for each edge segment in the image is selected. This process is performed for each edge segment in the left and right images. Figure 3 shows all the edge segments that have consistent left-to-right and right-to-left matches. Most matches at this stage are correct matches.

In the final matching stage, we use consistent local matches to form higher-level hypotheses on extended edges and junctions that have more global contexts. Within each context, maximum global consistency is used as a criterion to correct incorrect local matches. This is currently implemented by summing up the strength of each supporting local matches. The most likely match is then selected and enforces global consistency to correct inconsistent local matches in the same context. This process does improve the results of the second matching stage. We are currently developing more sophisticated global consistency matching technique that examines the justifications of local matches.

CONCLUSIONS

In this paper, we presented a new stereo matching technique based on geometric constraints. Edge segment is used because it has more features to compare and is more stable compared to individual edge pixels. In addition, combinatorial search has been significantly reduced in the matching process. Constraints are applied locally and the most likely match is selected based on how well the constraints fit the data. Because we use no assumption about the order of edge segment correspondence, this technique can potentially deal with more difficult stereo matching cases. We are currently working on more sophisticated global consistency matching techniques for better stereo matching.

REFERENCES

[1]. Arnold, David R., "Automated Stereo Perception," Stanford Artificial Intelligence Laboratory, AIM-351, March 1983.

[2]. Baker, Henry Harlyn, "Depth from Edge and Intensity Based Stereo," Stanford Artificial Intelligence Laboratory, AIM-347, September 1982.

[3]. Grimson, W. E. W., "From Images to Surfaces: A Computational Study of the Human Early Visual System," MIT Press, Cambridge, Massachusetts, 1981.

[4]. Marr, D. and Poggio, T., "A Computational Theory of Human Stereo Vision," *Proc. R. Soc. London*, pp. 301-328, 1979.

[5]. Moravec, Hans P., "Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover," Stanford Artificial Intelligence Laboratory, AIM-340, Sept. 1980.

[6]. Ohta, Y., and Kanade, T., "Stereo by Intra- and Inter-Scanline Search Using Dynamic Programming," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, March 1985, pp. 139-154.

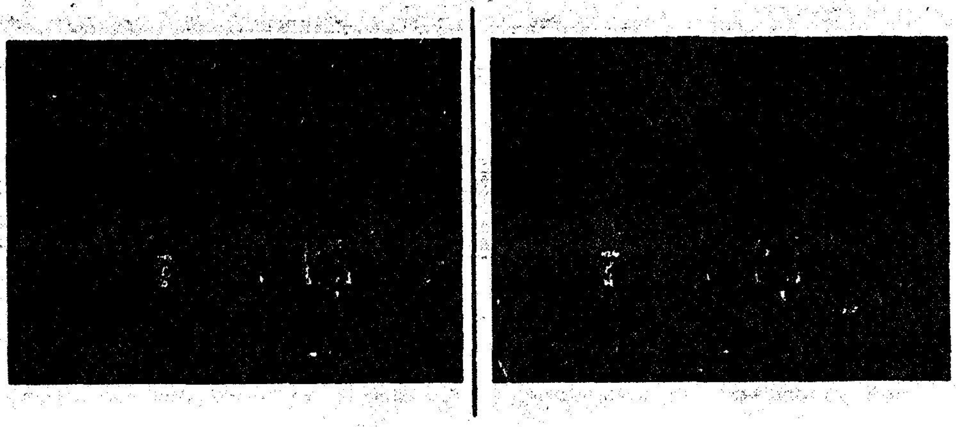


Figure 1: Stereo images.

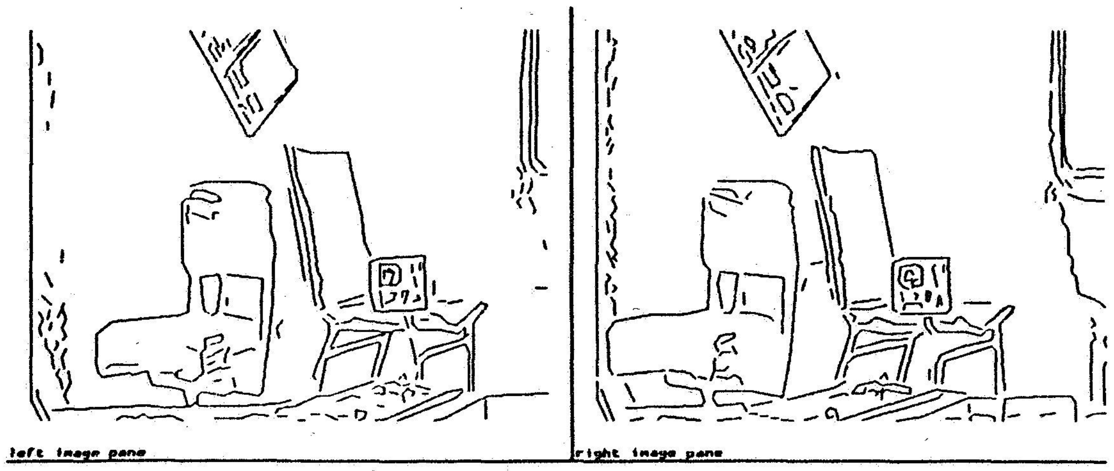


Figure 2: Edge segments in Figure 1.

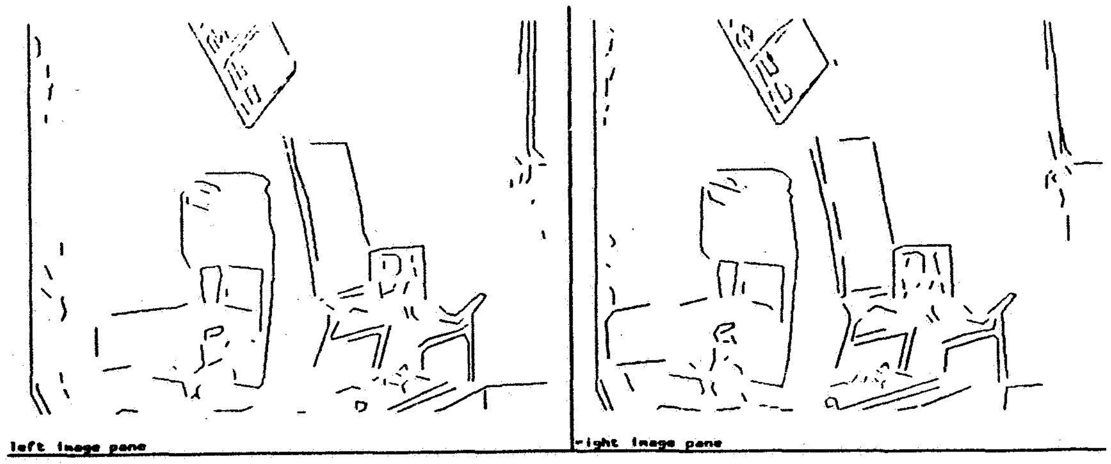


Figure 3: The final consistently matched edges in both stereo images.

S17-63
98
188137

A Database/Knowledge Structure for a Robotics Vision System

D.W. Dearholt and N.N. Gonzales
New Mexico State University
Las Cruces, NM 88003

N X 736581

ABSTRACT

Desirable properties of robotics vision database systems are given, and structures which possess properties appropriate for some aspects of such database systems are examined. Included in the structures discussed is a family of networks in which link membership is determined by measures of proximity between pairs of the entities stored in the database. This type of network is shown to have properties which guarantee that the search for a matching feature vector is monotonic. That is, the database can be searched with no backtracking, if there is a feature vector in the database which matches the feature vector of the external entity which is to be identified. The construction of the database is discussed, and the search procedure is presented. A section on the support provided by the database for description of the decision-making processes and the search path is also included.

I. Introduction

Several structures have been proposed which have properties desirable for use in a robotics vision database system. Some of these structures are examined in this paper, including a new family of networks in which link membership is determined by measures of proximity between pairs of entities represented by nodes in the database and the triangle inequality.

Suitable domains for the database structures considered here are those for which the entities to be stored are describable by a few feature vectors, e.g. color, or shape using Fourier descriptors. We consider the following to be desirable properties for such a database:

1. The database system should support efficient search, so that the feature vector which provides the best match to some external entity can be found quickly;

2. The structure should support classification, so that external entities can be named and higher levels of abstraction are supported;

3. The structure should support a modest level of self-description, so that entities along a search path provide information about the template-matching and classification decisions being made;

4. The neighbors of an entity should reflect consistency with respect to class, so that entities within a given mode of a class should be stored in a manner that reflects the associations and enhances retrieval;

5. Learning, considered to be the addition of entities to the database system, should be done in such a way that the previous properties are preserved.

While this list is too demanding to be well satisfied by any structure known to the authors, it is informative to explore the limitations of the various structures. The paradigm of preprocessing the data (entities to be represented in the database) so that search is facilitated is an important concept, selected by Dobkin and Lipton [1], Bentley and Friedman [2], and Bentley and Maurer [3]. Dobkin and Lipton [1] extended binary search to multidimensional search problems, and could efficiently respond to queries which included the nearest-neighbor problem. Bentley and his colleagues specialized in range searching queries, in which it is desired to find all entries in the database in which each component of the feature vectors is within some given range.

The k-d trees and range trees discussed by Bentley and Friedman [2] are interesting structures, designed specifically for range queries. Range queries are, of course, extremely useful for a wide variety of applications, and can be used as a sort of de facto classification scheme. Other papers presenting results of studies using k-d trees or range trees are Bentley and Maurer [3] and Chang and Fu [4]. These methods suffer the limitations imposed by any hierarchical scheme in terms of descriptive power and classification strategies, since abstractions are necessarily limited to those representable by hierarchies. Category information is, in fact, not a strong point of these methods, since only hierarchical neighbors of matching entities are readily available.

Kalvin, et al. [5], have developed a technique for pattern recognition based on geometric descriptions of the boundaries of objects. This technique is designed specifically for identification of overlapping and partially occluded objects. Attributes used for matching are derived from geometric features of segments of the boundaries of a set of objects. The search procedure is based on geometric hashing of objects in 5-space, where the coordinates are attribute values obtained during preprocessing of the data. A set of candidate matches (models) is selected on the basis of frequency of inclusion in the hypercubes of 5-space in which the unknown's attributes place it. A match rate is computed for each candidate match (model) based on the ratio of the number of match points for the model and the number of possible match points for a particular unknown. The database organization provides for efficient search for the specific application for which it was intended; however, no categorical information is provided and varying levels of abstraction are not supported.

Pathfinder networks share some attributes and objectives with memory-based reasoning, as discussed in Stanfill and Waltz [6]. Both paradigms make use of feature values to compute distance or dissimilarity functions, search memory for best match(es), and classify entities; and both paradigms share the philosophy of classifying entities by direct reference to memory. Pathfinder networks, however, are organized algorithmically so that associations are explicit, which results in categories being evident in the link structure. The search procedure described in Section III of this paper guarantees that search is monotonic (i.e., there is no backtracking) if there is a match in the database, so that search is not exhaustive, as it is in the scheme used for the Connection Machine described in Stanfill and Waltz [6]. Furthermore, the memory-based reasoning paradigm does not support descriptions of the search path and the decisions made which contribute to classification.

"Description" is used here to mean that the salient feature values of entities along the search path, the search path itself, the reasons for selecting the search path, and the neighborhood of the goal node, are available for a summary of the entire process. Section IV of this paper is devoted to a discussion of descriptive processes supported by the database developed here.

The organization of the entities in the database is based on a network model of semantic memory in humans. This model is called Pathfinder, and the properties of Pathfinder networks (PFNET's), previously called link-weighted networks (LWNs), are described in Dearholt, Schvaneveldt, and Durso [7] and in Schvaneveldt, Dearholt, and Durso [8]. Earlier work on databases intended for vision systems is described in Dearholt, Gonzales, Ellington, and Phillips [9], and in Dearholt, Gonzales, and Kirpekar [10]. These database schemas also used Pathfinder networks. Motivations for the database described in the latter paper and extended in this paper include (1) increased efficiency in the search process by eliminating backtracking; (2) the efficient determination whether or not a given entity is represented in the database; (3) organization of the database so that similar entities are clustered together; (4) provision for category-level information by means of the clustering inherent in Pathfinder networks; and (5) support of description of the search and classification processes.

The efficiency in the search process is accomplished at the time the network is generated by establishing links which provide a path between any two nodes so that the relative distance between nodes, as the path is traversed, is monotonically decreasing. Then, if a feature vector representing an external entity is presented to the database as a query, the corresponding node can be found rapidly from any node in the database. Heuristics to improve the initial node of the search can further improve the search efficiency. The determination that a given entity is not in the database follows from the procedure to be described in Section III. The clustering of similar entities and the resultant category-level information is a feature of the PFNET's to be described in Section II. These features provide a basis for the support of the description processes to be discussed in Section IV.

II. The Generation of a Pathfinder Network Database

Because PFNET's provide for clustering of similar entities, they seem to be a good paradigm for a database organization; indeed, their original purpose in modeling the semantic memory of humans provides for a database of concepts. Thus it seemed natural to extend PFNET's to feature-based applications in which each entity is described by a feature vector. Vision systems used for pattern recognition and image analysis are well served by such databases, and the properties of PFNETs support search, classification, and description, as mentioned previously. Our first effort in this direction was the database for insect identification (Dearholt, et al., [9]), in which PFNET's were used to organize the database. PFNET(∞, n-1) was used because it is the PFNET with fewest links, but there was no very effective search procedure associated with this PFNET organization. Our second effort (Dearholt, et al., [10]) justified and described the construction of the PFNET's which guarantee monotonic search. The purposes of this paper are to list the desirable properties of a vision database for robotics, and to describe the results of our work with Pathfinder networks relevant to these properties. It should thus be regarded as a progress report of our project, written for the purposes of the workshop.

The database schema we will present relies on the Pathfinder network model as a means of organizing the entities in the database. Development of this model (previously called link-weighted networks) has been ongoing for the past six years. Pathfinder yields network structures (PFNETs) for a set of entities, given estimates or measures of the pairwise distances between the entities. The original purpose of Pathfinder models was to model human semantic memory, so that the estimates of distances were typically estimates of similarity. For the database schema discussed here, however, the entities are each represented by a feature vector, and distances between pairs of entities are presented to the system as a weight matrix. If the weight matrix is symmetric, then the PFNETs derived from it are undirected, whereas an asymmetric weight matrix yields directed PFNETs.

A maximally connected network contains a link between every pair of nodes, so that each weight in the weight matrix is represented by a link. Such a network contains all the original information in the data, but it provides very little information about the structure underlying the data. Pathfinder includes only the links necessary to preserve geodetic paths, thus facilitating analysis and interpretation. Two parameters are required for the complete definition of a PFNET for a particular weight matrix. These are the r-metric and the q-parameter. The r-metric is the value of the Minkowski parameter which is used to compute the distance between nodes in the network which are not directly linked. That is, the weights along the path used to compute distance are individually taken to the r power, these values are summed, and the rth root of the resulting sum is the distance. In general,

$$d(N_i,N_j)=[\sum_{k=1}^{k} w_k{}^r]^{1/r}$$

where the w_k are the weights along the path between N_i and N_j. The r-metric parameter may take on values from 1 to ∞. The q-parameter determines the maximum number of links in paths considered to connect two nodes. For example, for $q = 2$, paths having more than two links are not considered in the preservation of minimum-distance (geodetic) paths in the PFNET.

PFNETs possess properties of inclusion which vary as values of the r-metric and q-parameter change (Dearholt, et al., [7]). Briefly, for a particular weight matrix, PFNET2 is a spanning subgraph of PFNET1 if and only if the r-metric used for PFNET1 is less than or equal to the r-metric used for PFNET2, provided that the q-parameter is held constant. In addition, for a particular weight matrix, PFNET2 is a spanning subgraph of PFNET1 if and only if the value of the q-parameter used for PFNET1 is less than or equal to the value of the q-parameter used for PFNET2, provided that the r-metric is held constant. The PFNET generated with $r = \infty$ and $q =$ the-number-of-nodes-less-one always has the minimum number of links and is the union of all minimum cost spanning trees of an undirected PFNET.

As a PFNET is constructed, precedence is given to small weight values, because they represent the strongest associations. During each stage of development of an undirected PFNET, the complete set of nodes is partitioned into connected subgraphs, called node sublists. When a link is added which joins nodes in different sublists, the two sublists are merged to form a single node sublist. Links in an undirected PFNET are labeled according to the basis for their inclusion in the PFNET. The four types of link labels are PRIMARY, SECONDARY_A, SECONDARY_B, and TERTIARY. A PRIMARY link provides the only path between a node sublist containing a single node and some other node sublist. A SECONDARY link joins two sublists which are not connected, and in which there are either alternate paths to terminator nodes, or the node size of both node sublists exceeds one. SECONDARY_A links are in all minimum cost spanning trees. SECONDARY_B links are in only some minimum cost spanning trees, as they provide alternate paths of the same length between two nodes. A TERTIARY link joins nodes within a single node sublist. TERTIARY links are not in any minimum cost spanning tree. The link-labeling rule yields important structural information, and the potential use of link labels in the descriptive processes will be discussed in Section IV.

Investigation of transformations on the values of the weight matrix has yielded two results of importance relating to the structure of PFNETs. A multiplicative transformation applied to the elements of a weight matrix preserves link structure in the PFNET for any values of r or q. A monotonic transformation applied to data in a weight matrix preserves the structure of the PFNET only for $r = \infty$.

The construction of the database for vision applications presumes a set of entities and some procedure to derive feature vectors to represent these entities. Typically, the entities to be represented by nodes in the database are examined to obtain salient features. Each class of feature values is presented as a vector; e.g., a color descriptor could include intensity values obtained from red, green, and blue filtered images. Similarly, shape descriptors might consist of a vector of Fourier coefficients. Difference measures for this paper are obtained using the $L1$ norm (the computation is the same as the Minkowski distance for $r = 1$) for each pair of entities, to obtain the weight matrix for input to Pathfinder.

The Pathfinder model preserves all geodetic (minimum cost) paths having no more links than the value of the q parameter, and leads to clustering based upon similarity of nodes. Pairs of nodes which are not directly linked in a PFNET are likely to be in different categories or subcategories. PFNETs provide a means of scaling data similar in some respects to clustering methods (e.g., Shepard and Arabie, [11]) and to multidimensional scaling (Kruskal, [12]), but the links in PFNETs provide information not directly available in clustering or in multidimensional scaling. Another network scaling scheme is NETSCAL (Hutchinson, [13]), but unfortunately Hutchinson did not consider triangle inequalities of dimension greater than two.

The domain assumed for the database consists of those problems in which each entity in the database is represented by a vector having d feature values, and corresponding features have their feature values in corresponding locations in the feature vectors. In addition, we assume that the features of the entities are such that taking the difference of corresponding feature values (as a part of applying the $L1$ norm) is appropriate. To begin the process of generating a PFNET, it is necessary to compute a scalar weight matrix W For the purposes of this paper, we will compute the scalar weight values of W using the $L1$ norm. For the Lm norm,

$$w_{ij} = [\sum_{k=1}^{d} |w_{i_k} - w_{j_k}|^m]^{1/m}$$

The $L1$ norm sums the magnitudes of the differences between corresponding components for the feature vectors being compared. If the $L2$ norm were used, it would be the Euclidean metric. For a discussion of distance measures suitable for data such as this, see Tversky and Krantz [14]. This article also justifies the Lm norms (the Minkowski metrics) as being the only metrics which possess both intradimensional subtractivity and interdimensional additivity, a feature that seems as important for vision databases as for cognitive modeling.

Although the r-metric parameter used with PFNETS can vary from 1 through ∞, for the purposes of the databases described in this paper, ∞ will be used. The consequence of this is that the distance between nodes not directly linked is the value of the largest weight along the path connecting the nodes. This is sometimes called the *dominant* metric. The databases in this paper use $q = 2$. The notation used for a PFNET in which feature vectors are used to compute W is PFNET(Lm, r, q), with the parameters in parentheses corresponding to the parameters discussed above.

A primary purpose of the databases constructed as PFNETs is to support effective search, so that notation for search paths is helpful. Search paths begin at some initial node, follow links established in the construction of the PFNET, and end at a node. Since there is at most one link between any two nodes, we will denote a search path from node N_i to node N_k (passing through N_j) as

$$P(N_i, N_j, \ldots, N_k)$$

N_i is said to be a *predecessor* of N_j because it precedes N_j in the search path. One way of viewing the network is to think of the entities N_i as points in d-dimensional space, establishing links according to the link membership rule of PFNETs, and traversing these links according to the search procedure to be described.

Another concept of importance is the *lune* of two nodes. The lune is discussed in Toussaint [15] in his definition and discussion of relative neighborhood graphs (RNGs). Lunes are also discussed in Lee [16] and in Katajainen and Nevalainen [17]. The lune of two nodes (points) N_i and N_j will be denoted by lune(N_i, N_j), and is defined as the set of points in which each point has a distance (we'll use the $L1$ norm, rather than the $L2$ norm as in the original work on RNGs) from both N_i and N_j less than the distance between N_i and N_j. In the weight matrix W, these internode distances using the $L1$ norm are already computed. Using $L1$, lune(N_i, N_j) is a rectilinear figure of dimension d. If $L2$ were used, lune(N_i, N_j) would be the set of points in the intersection of two hyperspheres of dimension d.

A notable difference between RNGs and PFNETs is in the assumptions usually made about the input spaces. For the RNG, two-dimensional space is normally assumed for the input data, but no such constraint is necessary for PFNETs. If the input spaces are two-dimensional, however, then the link membership of an RNG using $L2$ is such that the RNG is a spanning subgraph of PFNET($L2$, $r=2$, $q=2$). In the RNG, two nodes N_i and N_j are linked directly if and only if there is no other node in the lune(N_i, N_j). Euclidean distance is used in the 2-space in which the entities are customarily represented as vectors for the RNG, so that the lunes are intersections of circles.

Although PFNET($L2$, $r=2$, $q=2$) is satisfactory for a database organization, in terms of search for a matching entity in the database, the use of PFNET($L1$, ∞, 2) is preferable because the latter requires less computation in both the construction of the network and in search. Using $L1$, and assuming two dimensional input space, the link membership for an RNG is the same as the link membership of PFNET($L1$, $r=\infty$, $q=2$). For either, the link membership rule is

l_{ij} is in PFNET($L1$, ∞, 2) if and only if

$$w_{ij} \leq \min [\max [w_{ik}, w_{kj}]]$$

over all two-link paths between N_i and N_j.

This definition can be viewed as providing a link membership rule for one of a family of PFNETs, or as an extension of the RNG to a new application making use of the $L1$ metric.

The reason this PFNET is efficient in searching for a matching entity in the database is that the link placement is such that backtracking is never needed, provided that there is a matching node in the database.

III. Monotonic Search of a Pathfinder Database

Efficient, monotonic search, in which there is always a link to direct the search path(s) toward a node matching the external entity to be identified, is one of the attributes of a database organized as a Pathfinder network. Justification of monotonic search of a PFNET($L1$, ∞, 2) database follows. Consider that a set of entities N_i has been established with their corresponding feature vectors, and that the scalar weight matrix W has been computed using the $L1$ norm. Suppose that the PFNET($L1$, ∞, 2) has been constructed, and that E_x is an external entity represented by a feature vector compatible with the feature vectors of the N_i in the database; it is desired to find the N_k which provides the closest match to E_x within the database. Further suppose that the initial node (the node where search is initiated) is chosen to be N_i, and that node N_k in the database is a match for E_x.

That is, we assume for this discussion that the feature vectors for E_x and N_k are identical. The goal is to find a path between N_i and N_k, applying the match criterion at each node along the search path, until it is determined that E_x does indeed match N_k. An appealing argument can be made through the use of the lunes defined by the nodes along the search path. Consider lune(N_i, N_k) --if there is no other node in this lune, then N_i and N_k are linked directly, and a one-link search path connects the initial node with the goal node. Alternatively, if there is another node in lune(N_i, N_k), then N_i and N_k are not directly linked. But each node in the lune(N_i, N_k) is closer to N_k than N_i is to N_k, so that in progressing to any node in the interior of the lune, we diminish the distance to N_k. The node N_j closest to N_i will be linked to N_i, and the search path can proceed to N_j. The search path can, however, proceed to any node in lune(N_i, N_k) which is linked to N_i, and it is usually advantageous to go to the node which diminishes the distance to the goal node the most. Suppose the search progresses to node N_j. Here, the process is repeated with lune(N_j, N_k), and every node in this lune is closer to N_k than is N_j, so again the distance to the goal diminishes. In this fashion, the goal node is reached using only distance measurements between E_x and the nodes which are candidate successors for nodes on the search path, since we assume that E_x and N_k have the same feature vectors. That is, at node N_j, the difference between E_x and nodes linked to N_j is taken, and the difference which is smallest determines the next node in the search path. Thus the link structure guarantees that no backtracking is ever necessary if the entity E_x is in the database. If the distance from some node in the search path to E_x does not reach zero and cannot be diminished, then this indicates that there is no node which exactly matches E_x in the database.

A matching criterion based upon network properties is under development, although some aspects of a match criterion are necessarily dependent upon the problem domain. Throughout this paper, we presume that the matching criterion requires the goal node and E_x to match much more closely than E_x matches any other node in the database. Refinement of the theory of matching criteria is an area we are continuing to investigate.

There are four aspects of the search process for a database as described above, although the fourth is not always needed. These are:

(1) The selection of an initial node from which to begin the search, usually by means of some heuristic.

(2) The selection of a path from the initial node to the best matching node in the database.

(3) The application of the match criterion to each node along the path through the database, to determine whether any node on the path is a satisfactory match for the feature vector representing the entity to be identified.

(4) The determination of nearest neighbors of the node most nearly matching the external entity. Pathfinder networks support this search for nearest neighbors, because the link structure preserves geodetic (minimum) distances throughout the network.

The selection of an initial node can be accomplished by means of an index on some of the most salient features, so that, most of the time, the initial node is in the same class as the entity to be identified. Search efficiency is enhanced, of course, if the search is begun in the proper category. But the property of PFNET($L1,\infty,2$) of guaranteeing that from each node, the distance to every other node in the database can be diminished by traversing some link assures that the choice of the initial node does not affect the convergence to the goal node if the latter is in the database. This is important because it is not alway possible to begin the search at a node in the same category as the goal node.

The search procedure consists of the following steps, to be done at each node in the search path, from the initial node to the node at which the decision regarding a match can be made:

(1) The match criterion is applied to the present node (starting with the initial node) to determine whether or not the present node is a satisfactory match with the entity E_x. If the match is satisfactory, then halt.

(2) The distance $d(N_i,E_x)$ between each node N_i adjacent (linked) to the present node and the entity E_x is computed using the $L1$ metric.

(3) The node N_i which decreases the distance $d(N_i,E_x)$ the most is selected as the next node in the search path. If it is not possible to decrease the distance to the goal node (represented by E_x) then there is no matching node in the database. Otherwise, return to step one.

As an example, consider the set of nodes

$$\begin{array}{ll} N_1 = (2, 1) & N_5 = (9, 4) \\ N_2 = (4, 1) & N_6 = (9, 6) \\ N_3 = (5, 4) & N_7 = (7, 8) \\ N_4 = (3, 5) & N_8 = (10, 8) \end{array}$$

The weight matrix for this set of feature vectors, computed using the $L1$ norm, is

$$W = \begin{bmatrix} 0 & 2 & 6 & 5 & 10 & 12 & 12 & 15 \\ & 0 & 4 & 5 & 8 & 10 & 10 & 13 \\ & & 0 & 3 & 4 & 6 & 6 & 9 \\ & & & 0 & 7 & 7 & 7 & 10 \\ & & & & 0 & 2 & 6 & 5 \\ & & & & & 0 & 4 & 3 \\ & & & & & & 0 & 3 \\ & & & & & & & 0 \end{bmatrix}$$

The PFNET($L1,\infty,2$) is constructed using W, and is shown in Figure 1.

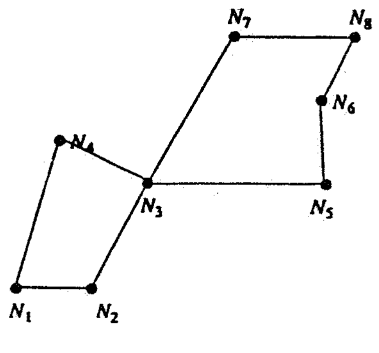


Figure 1

If the search is started at $N_1 = (2, 1)$, with $E_x = (10, 8) = N_8$, then the search path is $P(N_1, N_4, N_3, N_7, N_8)$, and the search procedure halts at N_8 with a match. Links are followed at each step, and the distance to the goal node decreases monotonically at each node.

The match criterion necessarily has some aspects which are domain dependent, but could be as simple as requiring that the distance between the goal node (providing the presumed best match) and E_x be less than some threshold value computed from the smallest distances between nodes in the database. Refinements could include the addition of an element of context in the sense that each category may have somewhat differing variability associated with satisfac- tory matching. The search process outlined above does not guarantee that the path with fewest links will be found, but it does guarantee that a path to a matching node will be found in which the distance decreases monotonically along the search path. If there is no node in the database which is an exact match to E_x, this is determined when the distance from a node N_j (on the search path) to E_x is larger than the distance from the preceding node in the search path to E_x. In this case, using the $L1$ norm, then it is not possible to get a better match to E_x than the predecessor of N_j; if that is not a satisfactory match, then E_x is considered not to be in the database. Formal proof of this is forthcoming in Dearholt [18].

IV. Description of Decisions and Neighborhoods

For an intelligent system, it is desirable to have support for the description of decisions made during the search process. This information can be very useful for communicating with the system in an attempt to understand not only the classification decision for a particular entity, but the properties of the neighborhood surrounding the entity in the network. The latter information can, of course, be used in some of the more sophisticated classification algorithms. Because of the clustering properties of PFNETs, and the directness of the search process associated with PFNET($L1$, ∞, 2), there is substantial information available regarding the classification results. The categories of nodes along the search path, and values of some of their most salient features, are the principal pieces of information used for our descriptive processes. We will focus mainly on four issues:

(1) The node where search is initiated,

(2) The search path,

(3) Link labels,

(4) The classification decision.

The selection of the initial node for the search procedure is a very significant decision. Although the PFNET($L1$, ∞, 2) guarantees convergence between any pair of nodes, the search time can be lessened substantially in a large database by judicious selection of the initial node. Beginning at a node which is in the same cluster as the goal node is a desirable objective; but the solution of this problem would imply that the classification problem for the entities in the domain is also solved. For many domains, the selection of a few key features which often lead to correct classification can be used to provide a sort of indexing into the network, so that the initial node could be the node having highest degree in the category indicated by the feature values in the set of key features. These key feature values, and the heuristic selection of an initial node based on them, are thus a part of the descriptive process at the beginning of the search.

As the search begins from the initial node (selected by some heuristic), at each node N_i in the search path some decision is made based upon the distance between the external entity and the nodes linked to N_i. The most suitable strategy, as discussed in Section III, is to select the node which most decr ıses the distance to the goal node, although there is no guarantee that the goal node will be reached in the fewest steps by using this strategy. The values of the feature vectors of the nodes which are candidate successors to N_i are available, and for the N_j which is the successor node to N_i, the feature value(s) which are most responsible for diminishing the distance to the goal are available for use in description. Together, these feature values are an indication of progress made toward the goal entity, since backtracking is never necessary. Furthermore, if there are multiple search paths (previously defined as the paths leading to the goal node), properties of all of the search paths can provide important information to the descriptive process.

The link labels on the links traversed also have some significance, as pointed out in Section II. PRIMARY and SECONDARY_B links are typically indicative that the search is progressing within a category, while the traversal of a TERTIARY link in a PFNET($L1$, ∞, 2) seems to indicate that the search path has progressed to a new category or subcategory. The traversal of a SECONDARY_A link also usually indicates entry into a new category. Proofs of these observations are difficult and not yet available, partly because the definition of "category" or "subcategory" is difficult. We are continuing to investigate the information provided by link labels, however, and that information is available for description also.

The last part of the search involves classification of the external entity, if that is possible. The choices of nodes available at the last step of the search provide information regarding the class and the centrality of the entity, so that some level of confidence in the decision of category could be assigned. That is, if the match with the goal node were borderline, and the node were near another category, then the confidence in the classification should not be very great. But if the goal node matched quite well and were also in the "center" of a category, then the confidence of the classification should be high. The Pathfinder paradigm supports a local search for nearest neighbors, so that this information can be used in either the classification decision or in the description of the neighborhood surrounding the goal node. The search for neighboring entities can be viewed as search by spreading activation, which would leave the goal node and travel to the neighboring nodes, so that their feature values and classification is available. Thus the centrality of the goal node, and its relationships to other nodes and categories, can be readily determined.

V. Summary and Conclusions

The construction of a database for vision applications using Pathfinder networks (PFNETs) was described, and it was shown that the search procedure associated with this database organization is monotonic (a distance measure steadily decreases at each node throughout the search path), provided there is a node in the database which matches the external entity. Relationships with the relative neighborhood graph (RNG) were discussed, and the search procedure was described. The database organization and search procedure provide a basis for descriptive processes of the decisions made along the entire search path, from the initial node to the goal node (or to the point where it is decided that there is no match in the database). Description based on these feature values and changes in feature values along the search path(s) and in the neighborhood of the goal node is expected to be useful in enhancing communication between the vision or robotics system and humans working with the system.

Research is continuing on some of the open questions encountered thus far. The match criterion used to determine whether the external entity matches some node in the search path "satisfactorily" is an important problem, and it has some domain-dependent properties and some characteristics which can be determined from a graph-theoretic perspective. The precise role of the PRIMARY, SECONDARY, and TERTIARY links in PFNET($L1$, ∞, 2) is also being studied, particularly how these link labels relate to category structure of the network. The descriptive processes are under investigation from the perspective of graph theory, although there seems to be a domain-dependent aspect also.

VI. Acknowledgements

The research described in this document was performed at New Mexico State University and was supported in part by National Science Foundation Grant IST-8506706 and by the Computing Research Laboratory, New Mexico State University.

VII. References

[1] Dobkin, D. and R.J. Lipton, "Multidimensional Searching Problems", SIAM Journal of Computing, Vol. 5, No. 2, pp. 181-186, June 1976.
[2] J.L. Bentley and J.H. Friedman, "Data Structures for Range Searching", Computing Surveys, Vol. 11, No. 4, December 1979.
[3] J.L. Bentley and H.A. Maurer, "Efficient Worst-Case Data Structures for Range Searching", Acta Informatica 13, pp. 155-168, 1980.
[4] J.M. Chang and K.S. Fu, "Extended k-d Tree Database Organization: A Dynamic Multiattribute Clustering Method", IEEE Transactions on Software Engineering, Vol. SE-7, No. 3, May 1981.
[5] A. Kalvin, E. Schonberg, J. Schwartz, and M. Sharir, "Two Dimensional Model Based Boundary Matching Using Footprints", New York University, Dept. of Computer Science, Courant Institute of Mathematical Sciences, Technical Report No. 162, Robotics Report No. 44, May 1985.
[6] C. Stanfill and D. Waltz, "Toward Memory-based Reasoning", Communications of the ACM, Vol. 29, Number 12, pp. 1213-1228, December 1986.
[7] D.W. Dearholt, R.W. Schvaneveldt, and F.T. Durso, "Properties of Networks Derived from Proximities", Computing Research Laboratory Memorandum 14, Computing Research Laboratory, New Mexico State University, MCCS-

85-14, June 1985.
[8] R.S. Schvaneveldt, D.W. Dearholt, and F.T. Durso, "Networks from Proximity Data", to appear in Computers and Mathematics with Applications, 1987.
[9] D.W. Dearholt, N. Gonzales, J. Ellington, and K. Phillips, "A Specialized Database for Classification Using Template Matching", Proc. of the Nineteenth Annual Asilomar Conference, pp. 494-496, Asilomar, California, November 1985.
[10] D.W. Dearholt, N.N. Gonzales, and H. Kirpekar, "A Vision Database Designed for Search, Classification, and Description", Proc. of the Twentieth Annual Asilomar Conference, Asilomar, California, November 1986.
[11] R.N. Shepard and P. Arabie, "Additive Clustering: Representation of Similarities as Combinations of Discrete Overlapping Properties", Psychological Review, 86, 87-123, 1979.
[12] J.B. Kruskal, "Multidimensional Scaling and Other Methods for Discovering Structure", in Enslein, Ralston, and Wilf (Eds.), Statistical Methods for Digital Computers. New York: Wiley, 1977.
[13] J.W. Hutchinson, "Network Representations of Psychological Relations", unpublished Ph.D. dissertation, Stanford University, 1981.
[14] A. Tversky and D. H. Krantz, "The Dimensional Representation and the Metric Structure of Similarity Data", Journal of Mathematical Psychology 7, pp. 572-596, 1970.
[15] G.T. Toussaint, "The Relative Neighbourhood Graph of a Finite Planar Set", Pattern Recognition, Vol. 12, pp. 261-268, 1980.
[16] D.T. Lee, "Relative Neighborhood Graphs in the L1-metric", Pattern Recognition, Vol. 18, No. 5, pp. 327-332, 1985.
[17] J. Katajainen and O. Nevalainen, "Computing Relative Neighbourhood Graphs in the Plane", Pattern Recognition, Vol. 19, No. 3, pp. 221-228, 1986.
[18] D.W. Dearholt, "Associative Network Databases Constructed for Efficient Search", in preparation.

518-63
- 9P.
188138

Sensing and Perception: Connectionist Approaches to "Subcognitive" Computing

J. Skrzypek
University of California, Los Angeles
Los Angeles, CA 90024

CD 146017

ABSTRACT

ABS ON P.152

The machine perception laboratory represents a new paradigm for research in Artificial Intelligence at the Computer Science Department of UCLA. It is based on synergistic intermixing of methods and knowledge from the fields of Artificial Intelligence and Neuroscience.

-o- *The Neuroscience is a source of fundamental concepts about function and mechanism of natural vision and perception; it motivates our view of inseparability between algorithms and neural substrate.*

-o- *The AI explores computational theories of vision and perceptual reasoning by inventing algorithms and implementing them as "connectionist" architectures.*

The underlying intent of this interdisciplinary approach is to transform scientific knowledge into an engineering form of a general purpose machine perception by viewing "neural" connections as a paradigm for parallel computations.

The future of intelligent robots depends on succesfull implementation of a robust perceptual system. Although many clever forms of robotic vision have been engineered, a general-purpose machine perception remains a distant goal. Computing architectures best suited for global perceptual function pose one type of a problem. Another problem stems from the limitations of sequential computing paradigm where the number of functions which naturally map onto Von Neumann architecture is restricted. In natural system, visual functions are supported by a variety of parallel structures. This motivates our belief that future advances in a general purpose perception must assume inseparability of *function* from *strucutre*.

Our prototypical computational architecture consists of hierarchically structured layers of processing units that perform dedicated functions. Both discrete and real-value passing architectures are considered. Physical representation of transduced stimuli is implemented as a well structured connectivity between "neurons" and the computations are performed by types and weights of different connections. More precisely the computation is a result of some process, realized as "neuronal" functions, that is applied to a spatio-temporal "image" of signals. The process and the constraints are embedded into our connectionist architecture. The translation to more abstract levels is done through aggregation of features by an interpreter, which in early vision may be implemented by fixed connections. The ultimate goal of this project is to conceptualize a computing structure which could eventually be implemented in hardware.

Acknowledgement

This project was supported in part by NSF grant #ECS8307553, ARCO fellowship #1. Laboratory computing environment was made possible by a generous gift from Hewlett Packard and IBM. Thanks go to MPL members, David Gungner, Edmond Mesrobian, Paul Lin, Inge Heisey, Mike Suber and Eugene Paik for their suggestions and criticisms.

I. COMPUTERS AND BRAINS -- MOTIVATION

This paper is divided into four sections. First we outline the intellectual needs for integrating the knowledge about percepuon in man and machine. The second section presents our notion of large grain architecture as a computational environment for studying global functions of machine perception. In the third part we describe the small grain architectures represented by "neural networks" that provide a computational substrate for perceptual functions. We conclude with architectural models of two early-vision operations implemented as neural networks that embody the *principle of inseparability between structure and function.*

a. Intellectual motivation

Intellectual motivations that unify studies of human and machine perception, - including vision, touch, proprioception, range and other sensory modalities, - derive from assumption that information processing is fundamental for intelligent behavior. Perception, spatial reasoning and learning are the attributes that will differentiate the next generation robots from present day automated manufacturing. The ultimate test for Artificial Intelligence is the invention of an autonomous mobile robots, whose "intelligent" behavior emerges from linking perception to motor output. Modern computer science plays a pivotal role in understanding information processing systems. On the other hand, mechanisms and functions of information processing underlying human intelligence are in the domain of Neurosciences. The rapid growth of these disciplines in recent years is advancing our understanding of perception. It is hoped that interdisciplinary combination of Artificial Intelligence and Cognitive Science will provide more rigorous, scientific fundations for this research.

What can be expected from a general theory of perception developed by such crossdisciplinary approach? In the short term it should help us understand how the elements of perception have evolved in natural systems and what are their limits. In the long run, a theory of perception should help us to formulate questions that extend beyond presently limited engineering knowledge of this function. For example, can we improve upon biological perception when implementing these functions in mobile robots? Is human perception limited by characteristics inherent only to biological systems? Are these limits imposed by algorithmic principles or by the underlying substrate? What is the grain of computing architecture most suitable for cognition and perception?

b. Perception and AI

Our working goal for Machine Perception and in particular for Computer Vision is a development of computing systems that can accomplish tasks previously only achieved with human intelligence (1). Discovery of heuristics used to constrain the problem according to physical laws should eventually lead to models of greater generality (2). In the past these efforts were strongly limited by the computational architectures available to the designer. The sequential computing paradigm limits solutions for computer vision that can operate in real time by restricting a selection of functions that naturally map onto Von Neuman architecture. In natural systems, visual functions are supported by a gamut of physical structures that are inherently masively parallel (3). Hence, we believe that further progress in realization of general purpose computer vision that operates in real time must be based on assumption that function and the underlying computational substrate are inseparable. The chances of success can be maximized by combining traditional, forward-engineering approach to synthesis of computer vision system with analytic viewpoint as characterized by Neurosciences where the intent is to reverse engineer the solution. This is difficult because the current knowledge about anatomy and physiology of neuronal networks underlying manipulation of mental imagery does not allow easy introspection on such processes at the level of subcognitive computation (4, 5). Nevertheless, models of mental computation underlying perception and cognition must be build and verified. Approximation of such tests at the present time, is possible only through computational models in the realm of AI (6). Our approach to studies of cognitive and perceptual functions is detailed in next section and it involves coarse grain architecture represented by networked AI workstations. On the other hand, the notion of local computation supported by fine grain architectures resembling neural networks is developed in the third chapter.

Perception may be thought of as an example of a continuous problem solving operation. It is an active process during which hypotheses are formed about the surrounding environment (see 7). Sensory information acquired through vision, touch, smell, sound and proprioception is integrated to evaluate these hypotheses (8). In each of the sensory modality analog data must be first acqured and preprocessed. This stage is similar to data driven signal processing operations that are well understood in the realm of Electrical Engineering. The next stage involves segmentation and labeling of the preprocessed sensory data (2,1). And the last stage involves understanding of the sensory information in every modality and integration for perceptual reasoning. This representational view of processing derives from generally accepted model of visual perception. Considering recent advances in computer based simulations we can implement in software any model of perception. A critical question is which mechanisms must be incorporated into hardware to guarantee human-like performance. Which architectures would make basic perceptual capabilities including learning and problem solving, feasible for autonomous mobile robots? The natural computation is based on different principles than those embodied in computers. It is a task oriented process where the current situation, including goals and drives, directly determine the next action (9). The human brain has many highly developed structures, dedicated to performing different functions, even though externally it appears to act as general-purpose system. Unravelling mysteries of perception and cognition is one of this century's major scientific challenges.

c. Neuronal architectures and parallel computation

The inspiration that AI derives from Neuroscience is based on assumption that manipulation of symbolic representations is fundamental to emergence of intelligence (2, 10). Hence, computers as symbol manipulating systems could allow us to create and test models of perception as computational activities of the brain. Since we are the keepers of information about this world we can construct the programs and data structures that internally to computer represent any concept that normally refers to external environment. The simulation running on a computer can perhaps be likened to cognitive processes that allow to reason about the consequences of physical actions before they take place. The central question is whether we could create an artificial symbolic system that uses sensory information to construct abstract representations of external world. If AI techniques will allow us to realize such symbolic behavior in a computer-based system will it have to be based on neural principles (9)? And if so how can we implement symbolic processing in terms of neural networks?

The desireability of neuronal architectures derives from massive parallelism (hence, real-time performance) and computation based on connectivity (hence, simplicity) (11, 12, 13). Parallel computation has recently become a major concern for computer science. The constraints of solid state physics limit further evolution of sequential machines to increasing speed via optical computing. And the developments in VLSI favor parallel architectures. To gain speed, one school within parallel computing paradigm assumes that computation can be performed by a pattern of connections between slow and simple processors (11, 12, 13).

Fine grain massively parallel architectures are similar to neuronal structures in the sense that they are based on millions of interacting processors. One of our immediate research problems is to investigate how can we realize such strucutres and how to compute with them. Because of close resemblance to anatomy of natural computing structures, this class of architectures might offer the most plausible solution to machine perception in real time (12, 14, 9, 15).

Past approaches to computer vision were based on the assumption that it can be solved in the abstract domain unrelated to the underlying physical mechanism (1, 16). Our approach differs because we constrain the problem by requiring a solution to be implementable in a 3-D connectionist architecture. The fundamental premise of *connectionism* is that individual neurons do not actively manipulate, large amounts of symbolic information (12). One of the major modes of information processing in the neural systems can be described in terms of the relative strengths of synaptic connections. Therefore, rather than using complex units that manipulate symbolic inputs, connectionist architectures computes by modulating signal with appropriately connected simple units. Hence, the computation is a form of cooperative/competitive relaxation process, taking place in a distributed net of "neural" elements.

Our approach is different from multilayer perceptrons because we propose that each unit has an S-shaped transfer characteristic (44), which can be modeled by: V = Vmax [X / (X+k)], where V is the output, Vmax is the saturating level of the output signal, X is an input and the k is the input value that generates the half maximal response. This is consistent with physiological evidence for saturating membrane response and distributed synaptic inputs. inputs. The sigmoidal function allows for automatic sensitivity control, computation of relative values in context of the neighborhood and others. Thus unlike the "binary" thresholding function in perceptrons, our networks will always operate in the most optimal configuration (17).

The "neuronal" operators can have thousands of inputs and tens of outputs. A continuous output value can be generated as a thresholded hyperbolic tangent function of weighted inputs. Weights allow us to implement both positive and negative averages. Presynaptic inhibition, dendro-dendritic synapses and the concept of relative changes carrying information completes the architectural environment. These elements, allow the implementation of convergence and divergence of signal pathways as well as lateral interactions between spatially distinct nodes. Simulation of specific computing architectures is supported by UCLA-PUNNS, a neural network simulator developed in my laboratory to address the question of inseparability of function and computing substrate (18).

Principles of computation behind our simulated model are inspired by the neurophysiology of interacting neurons (19):

---o--- Concurrent computation is supported by parallel active connections between neuronal-operators, arranged in a hierarchy of layers.

---o--- Computation is performed in the analog domain and can be simulated as real-value passing networks.
---o--- For early processing stages all intra and inter layer connections are fixed and control is executed by feedback pathways which selectively modulate activity in a single operators.
---o--- Adaptive properties of the networks derive from relaxation-like behaviour, computed by each layer at the multiple scales of resolution.
---o---The cooperative and competitive modes of relaxation are computed by agonistic and antagonistic lateral interactions between neuronal operators.
---o--- Connections are modeled by weights resembling synapses with sigmoidal input-output characteristic.
---o--- Abstractions at higher levels are defined by the specific architecture of connections.
---o--- Segmentation is partially determined via bottom-up linking of many simultaneous computed images of primitive attributes.

Our principal architectural module is a three-layer computing structure (18). The INPUT layer carries a topologically correct representation of the scene. The OUTPUT layer is an abstraction which does not have to be spatially indexed to the original image. Local constraints are built into the layers. Global and local constraints are computed by the CONTEXT layer. The advantage of our concept is that it is general enough to allow the implementation of parallel architecture for signal manipulation and for aggregation of feature maps in the symbolic domain.

d. Neural net representation of perceptual knowledge.

In Computer Vision systems, programs performing visual functions are constrained by the architecture. The robustness of the human perceptual system stems from its ability to adapt/program itself. Thus novel stimuli can be processed by newly developed computing structures. Plasticity itself does not explain perception, but ability to program new knowledge and to search for alternative hypotheses is fundamental to perceptual tasks. A priori knowledge of selection criterion will always allow to exhaustively search and find an optimal model that satisfies the postulated hypothesis. The question is however, can such solution and its alternatives be identified in a reasonable time. Hence, the need for massively parallel computation in a form of neural nets.

We know that knowledge allows to optimize the search process (1). This poses a question of how to organize and represent knowledge in a memory so that it can be easily accessed at the right time (20). The factual knowledge, as opposed to "how-to" knowledge, can be organized into *networks* of associations, so that access to one part provides *connections* to other relevant parts. The knowledge about the scene must include the specifics of visually perceived objects plus the knowledge about a variety of objects in all related scenes or functions. This suggests hierarchical, as well as associational, structure. How to realise such architecture with connectionist structure, how to map the relevant knowledge onto patterns of connections and how to make it program itself by changing connectivity without "forgetting" are some of the questions that we are facing.

Perceptual knowledge must incorporate world information derived from integration of different sensory modalities. "Nihil est in intellectu quod non sit prius in sensu" (St. Thomas of Aquinas 13c), there is nothing in our intellect that did not pass through our senses. Most of our knowledge about the environment comes to us through one of the five senses. Hence, understanding the workings of these systems is a prime scientific problem. This problem is magnified in the technological realm. Vision is indispensible for autonomous mobile robots, and there is some progress in this area. Other sensory modalities are more neglected, because it is not clear how to best use them and how to implement practical solutions. In general, a solution to sensory interactions with the environment is a precursor to adaptable, intelligent performance in for example, industrial settings or in space exploration (21). The problem of best architectures or environment for studying questions related to sensory integration is open. The key questions that must be addressed are transmodal equivalences, sensory-mode specific knowledge and constraints, merging of representations specific to modality, and disambiguating conflicting modal specific information. These problems represent an important scientific challenge to implementation of machine perception.

II. MACHINE PERCEPTION LABORATORY

The coarse grain architecture of the machine perception environment consists of four networked AI workstations, each performing dedicated function (fig.1). The vision station simulates the action of the "EYE" and some higher level visual functions. The "HAND" is a separate station that provides the environment for studying manipulation and locomotion in support of perceptual task. The Ethernet fulfills the role of the spinal cord by allowing to integrate other sensory modalities, such as range, proximity, touch, etc., controlled by the "SENSE" workstation. The fourth AI workstations simulates higher level cognitive functions of the "BRAIN". The ultimate goal of this evolving architecture is to build an environment where by experimenting with global functions of machine vision and perception we could reduce scientific concepts to engineering solutions.

Although a complete theory of perception is a distant goal, both machine intelligence and humans must acquire and manipulate information from the environment. Moreover, this information must be organized into a store of knowledge that can be applied to future problems. LISP-based environments offer many advantages for experimenting with issues related to highly adaptive, multisensory based robotic systems. Such integrated environments will allow us to approach problems of vision, sensory integration, assembly and inspection as general scientific issues of planning, perception, problem solving and spatial reasoning. The machine perception laboratory (MPL) offers a realistic experimental test-bed for developing and validating various hypothesis related to robotic perception and intelligence. It also allows us to integrate and evaluate different software packages dealing with perception and manipulation.

Some software systems and tools are available currently either in academic or industrial environment, that could enhance performance and eliminate the expense of rediscovery. Of course this is feasible only if there is an environment which easily allows integration of existing systems. These packages include, among others, systems in vision, planning, decision-making, data fusion, reasoning, problem solving etc. The MPL, including all hardware/software systems, is in a continuous state of evolution and offers a diversified experimental environment spanning fields of computer science, artificial intelligence, robotics and cognitive sciences.

a. System organization

The system can be seen as a hierarchical organization of separate processes running on different workstations (22). Each dedicated station is a complete LISP-based environment extended with functions and procedures appropriate for experimenting in its domain. Part of the integration issue is addressed by extending the total environment with functions that accept, interpret and execute commands emanating from a dedicated station called "BRAIN" and send back results of their domain-specific computations. In this sense the dedicated station behaves as a lower-level entity, capable of understanding high-level commands and executing them by triggering specialized procedures appropriate to the task.

Such stations perform multitasking operations in their domains while at the same time running under the multitasking environment of the "BRAIN". This will ease the TOP-DOWN integration of the system since programming will be limited to writing specialized functions for the brain. Message-passing programming, inherent in an advanced AI environment will ease inter-task communication and the integration of new workstations. At the same time, the implementation of the system as a network of independent station will preserve their integrity, support parallel execution (vision and touch), and perhaps allow for easy integration of software written in other languages.

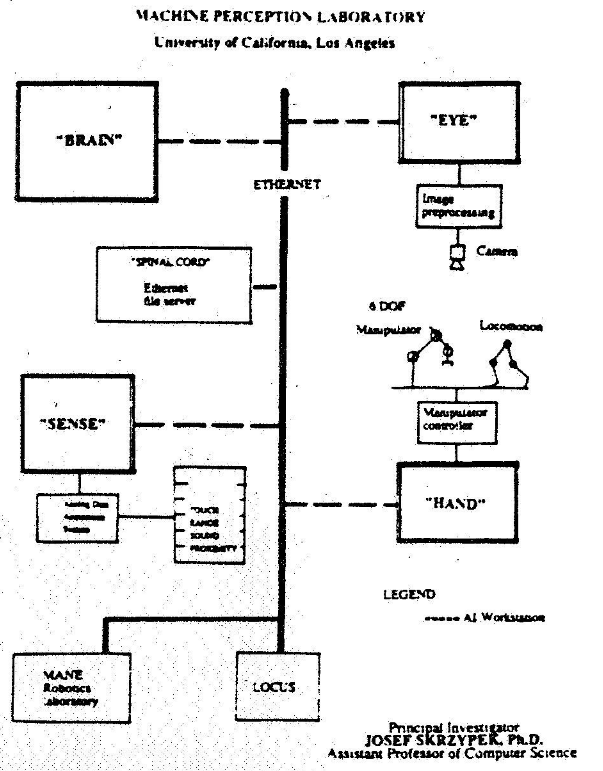


Figure 1. Global computing environment for the Machine Perception Laboratory

This implementation assumes that stations are loosely coupled. It is not our intention to study problems of close interactions between subsystems or patterns of data flow and performance in real-time. It is intended however, that the major portion of computation inherent to a specific domain, for example vision, will be performed on dedicated station called "EYE".

b. Integration

One of the key problems in setting up MPL is integration. Issues of synchronization, programmability, communication, load balancing, parallel execution are all nontrivial problems and could be analyzed by estabilished areas of computer science. We envision that MPL will consist of a few, networked AI-based workstations and dedicated computers. Because of this, a unified LISP environment will aleviate many problems inherent in integration of such complex systems. One of the key issues of integration will be to combine symbolic and numeric computation in each sensory modality. An example of successful solution to this problem in the area of vision is given in (23).

Our initial research in the perceptual functions is focused on integration of vision with other sensory modalities. Hence, the notion of multiple networked AI-workstations, each dedicated to separate perceptual function. The LISP environment provides tools for easy integration of separate processes operating on different work stations in the network. Additionally, it allows for easier incoropration of software modules written in other languages.

The "BRAIN" plays the role of organizing problems at the task level and it assumes the responsibility of distributing computing to proper stations. Using a LISP environment to implement "BRAIN" facilitates and enhances its performance. It is relatively easy to create facilities for programming functions that can request services of remote procedures, gather high-level information from different sensory modalities, and interrupt or activate processes such as manipulation running on the other stations.

Such an environment lends itself to incremental development and testing of complex perceptual behavior. Separately developed and tested sensory or manipulation operations can be integrated as primitive functions in the "BRAIN'S" repertiore. Task-level programming, world modeling, and manipulation of symbolically represented infromation is fundamental to implementation of cognitive functions (24).

III. UCLA PUNNS: NEURAL NET SIMULATOR

Previous section presented an example of a coarse grain architecture, most suitable for studying global functions of perception. In this part we focus on environment for studying neural networks as physical substrate underlying local computation in perception. Physical interactions with our world demand real-time responses. If a machine is to maneuver and operate in an underconstrained, natural environment, its efficacy and survivability will also depend on how quickly it can perceive and respond (25) Natural systems solved the problem of real-time constraints by using massively parallel neural networks. The capabilities of autonomous, mobile robot are restricted by the size, weight and power requirements of the computer (26). The amount of support that a computer extracts from the machine is one of the critical factors in determining the feasibility and functional capabilities of a system. The progress in this area may come from conceptually new architectures based on neuronal principles. Hence, the need for powerful simulation tools.

Despite numerous studies over the last fifty years, we don't have a satisfactory explanation of perceptual phenomena. Part of the problem stems from inability to describe the process. Von Neumann speculated that the structure and the state of the neural network might be the simplest way to describe perception (27). Our approach to machine perception is based on assumption that the network structure yields the function and, vice versa, that the real-time function of perception implies a particular neural network structure. This approach is motivated by the reductionist view of neurophysiology where the principal notion is to explain function in terms of structure (19).

To investigate the relationship between structure and function, we have developed PUNNS (Perception Using Neural Network Simulation). PUNNS (59) is a continuously evolving environment that allows to study the functionality of massively parallel computational structures as applied to image data. The initial focus is to study neural structures that allow execution of visual functions in constant time, regardless of the size and complexity of the image. Because of complexity and cost of building a neural net machine, a flexible neural net simulator is needed to invent, study and understand the behavior of complex vision algorithms. Some of the issues involved in building a simulator are how to compactly describe the interconnectivity of the neural network, how to input image data, how to program the neural network, and how to display the results of the network.

a. Neural simulators

The theoretical properties of pseudo neural networks as applied to logical computation, learning and adaptation have been extensively explored and reviewed elsewhere (27, 28, 29, 30, 31). Many of these approaches have nothing in common with neurophysiology. Nevertheless, they do indicate the diversity of behavior that results from the interconnection of simple computational elements. PABLO is an example of a simulator that provides precise

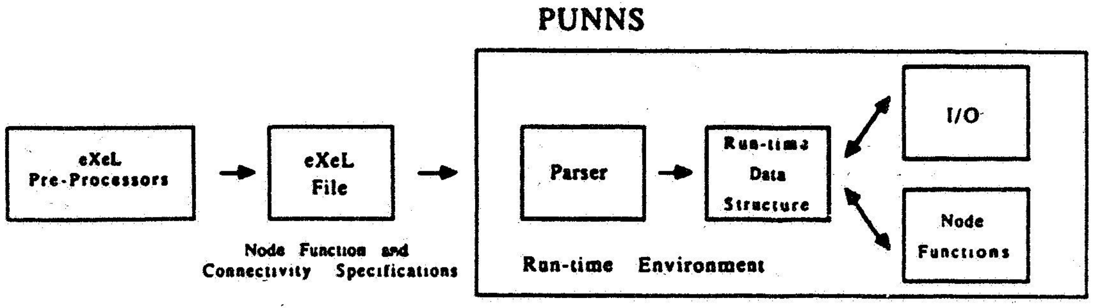


Figure 2. Block diagram of PUNNS run-time environment.

modeling of neurons and their interactions (32). Its environment closely supports many known properties of soma membrane, synaptic physiology, dendritic propagation, and axonal transmission. BOSS is another discrete-event simulator that was designed to investigate large neural networks (33). In contrast to PABLO, where each individual neuron was specified and interconnected, BOSS forms a statistical representation of the connectivity pattern. This allows for the relatively fast simulation of large connectivity patterns.

In contrast to these batch type simulators, ISCON offers the advantages of an interpreted simulator and network construction tool (34). It is written in LISP and it allows to dynamically change network connectivity and restart the simulation. The penalty for this flexibility is that large networks take prohibitively long to execute. To increase execution speed while maintaining flexibility, ISCON evolved into the Rochester Connectionist Simulator (35). RCS is a run-time environment written in C that allows user written programs to access a library of connectionist type functions, e.g. building networks, setting potentials, examining nodes.

b. PUNNS environment

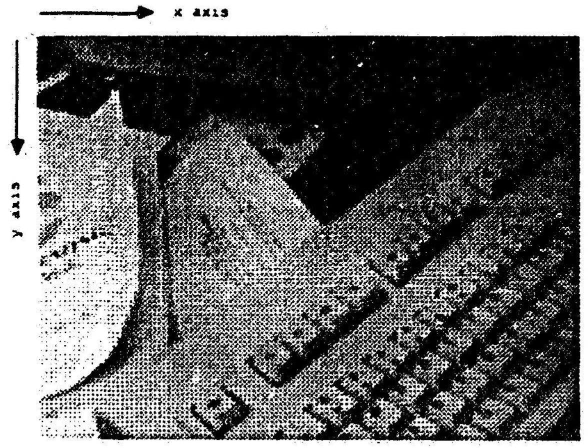
The run-time environment of PUNNS is fast and robust (fig. 2). PUNNS was implemented in C under System V and has been been ported to 4.3bsd. The underlying simulation approach used was a discrete time simulation technique that has each node visited at each simulation time step. This approach is especially useful when input data is changing every few time steps. A connectivity language (eXeL) was developed that describes the functionality of individual nodes and how they are interconnected. Complex connectivity patterns using large numbers of nodes can be generated by eXeL pre-processor routines. These are programs that output eXeL files. Hence, they are easy to modify when the connectivity pattern must be adjusted. After loading the eXeL file into PUNNS, the parser builds a data-structure which can be quickly interpreted to produce the simulation of the neural network. Changing node functions or connectivity is accomplished by reloading a modified eXeL file. Input and output to the simulation is done through graphics windows. Real images are used as a test data for the synthesized networks. A node's function can access a particular range of pixels from a graphics window and can display the result of a node, after firing, in an output window. Stimulus and response of a net can be displayed by using multiple windows. Activity levels in a layer can be viewed in one window, and the window can be saved as an image. This snapshot of activity can be then placed in an input window and newly loaded layers can continue processing from it.

In PUNNS, local connections and global mappings are used to separate the ideas of neighborhood node interactions and the connections established between functionally different blocks of nodes. Local connections are responsible for receptive field size and property, while global mappings may or may not be topologically preserving. A node's function tells what a node computes from its inputs and its temporal properties describe how the excitation level changes over time. The node is the lowest level primitive that represents an idealized, lumped parameter model of a neuron. Node description specifies inputs from other nodes input and the functions which are to act on these inputs. PUNNS also allows for dendritic input to a node, with each dendrite having a possibly unique processing function. All nodes are specified in eXeL files as follows (italics indicate a user definable parameter):

node *node-name*;
  initial-value, *length-of-history*;
  node-function;
  dendrite-1:*dendrite-function1*, *node-name16*, ... ;
  dendrite-2:*dendrite-function2*, *node-name38*, ... ;
  soma, *node-name23*, ... , *node-name42*.

The *initial-value* of the node allows a selection of different initial value. The *history-length* of a node indicates how many past excitation levels should be saved which is useful in modeling exponential decay. The *node-function* is implemented in C, it exists in the PUNNSrun-time environment and is fired when executed by the simulator. The *dendrite-function* performs the same purpose as the *node-function*. There is no provision for modeling a delay in dendritic propagation outside of synaptic transmission. The dynamic behavior of neural networks can be modulated with a *time-delay* option that is synonymous with multiple synaptic delays.

PUNNS has been used to model and simulate pre-attentive texture segmentation (36) and the generation of matching heuristics from time-varying images (18). Figure 3 illustrates how a conceptual structure, in this case a center-surround receptive field, is analyzed using PUNNS. The structure of this receptive field, forms a strongly excitatory center and a concentric inhibitory surround. When multiple, overlapping center-surround receptive fields are applied to an image (fig. 3a.), the result is a pattern of activity that highlights discontinuities in image intensities (fig. 3b). As the transition



(b)

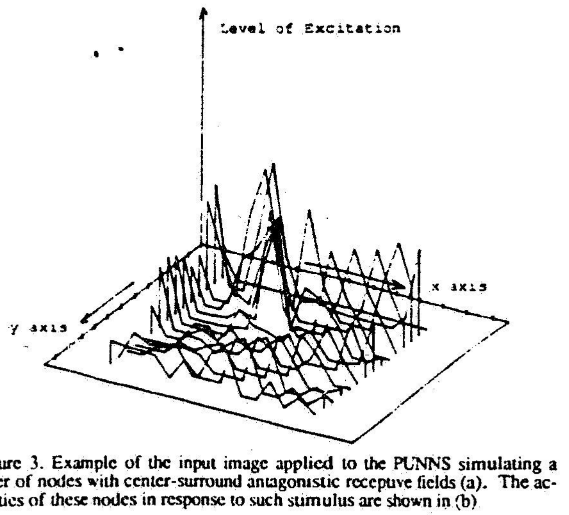


Figure 3. Example of the input image applied to the PUNNS simulating a layer of nodes with center-surround antagonistic receptive fields (a). The activities of these nodes in response to such stimulus are shown in (b)

in intensity becomes stronger, the node's excitation level increases. This structure was easily prototyped in the PUNNS environment and the simulation time was under thirty seconds.

IV. APPLICATIONS: VISION THROUGH CONNECTIONS

In this section we present examples of two early vision functions which have been implemented and analyzed using principles of neural networks.

1. Constancy preprocessor

The success of autonomous mobile robots depends on the ability to understand continuously changing scenery. Present techniques for analysis of images are not always suitable because in sequential paradigm, computation of visual functions based on absolute values of stimuli is inefficient. Important aspects of visual information are encoded in discontinuities of intensity, hence a representation in terms of relative values seems advantageous (2,3). This example deals with the computing architecture of a massively parallel vision module that optimizes the detection of relative intensity changes in space and time.

Visual information must remain constant despite the variation in the ambient light level or in the velocity of a target or a robot. Constancy can be achieved by normalizing motion and lightness scales. In both cases, basic computation involves a comparison of the center pixels with the context of surrounding values. Therefore, a similar computing architecture, composed of three functionally-different and hierarchically-arranged layers of overlapping operators, can be used for two integrated parts of the module. The first part maintains high sensitivity to spatial changes by reducing noise and normalizing the lightness scale. The result is used by the second part to maintain high sensitivity to temporal discontinuities and to compute relative motion information. Conceptually, the constraints and the rules of transformation are embedded into a computing structure which transforms the original image into two new representations. One carries the information about discontinuities in space while the other represents intensity changes in the time domain. This is consistent with the notion of space-time equivalence which suggests a hierarchical design where spatial normalization is performed before dealing with temporal domain.

Simulation results show that response of the module is proportional to contrast of the stimulus and remains constant over the whole domain of intensity. It is also proportional to velocity of motion limited to any small portion of the visual field. Uniform motion throughout the visual field results in constant response, independent of velocity. Spatial and temporal intensity changes are enhanced because computationally, the module resembles the behavior of a DOG function.

1a. Spatio-temporal considerations

Natural illumination can vary by ten logarithmic units of intensity. This exceeds the response range of artificial or biological sensors (3, 40). Hence, the first problem is how to maintain constant sensitivity to light changes over the whole intensity domain while preserving a "unique mapping" between the reflectance properties of the surfaces and perceptual notion of lightness. Linear variations of intensity usually are a consequence of nonuniform illumination (38) that can be filtered out without loosing meaningful information. The new representation of the image is expressed as relative values of intensities, that corresponds to spatial discontinuities generated by object boundaries. Absence of a DC component introduces a need for some reference point necessary to achieve lightness constancy.

Lightness constancy can be viewed as a problem of maintaining high sensitivity regardless of local or global ambient light level (39). This implies constant response when the illumination throughout the scene is multiplied by a constant. In addition, essential information such as edges must be preserved. One solution is to have sensors with a steep intensity-response (I-R) characteristic, spanning 3 log units of intensity and a mechanism that automatically shifts the operating curve to the prevailing ambient light level (40).

Nearby areas of a scene tend to have approximately equal illumination and reflectance. Hence, we use local intensity averages to set the upper and lower thresholds of the response curves. This is done automatically by adjusting the midpoints of the I-R characteristics to the local ambient light levels (40). Thereby invariance under local addition of linear illumination bias is achieved. Similar argument holds for global averages which in addition reduce sensitivity to noise by removing bias due to overall average illumination.

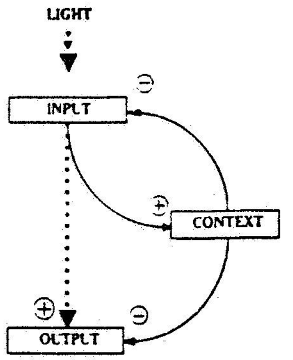
The detailed description of normalization is given in (17). Therefore briefly, this operation is performed by spatial operators with two antagonistic zones, center spot and surrounding annulus, better known as center/surround receptive fields (C/S-RF) (3, 41). The C/S uses lateral inhibition to emphasize contrast or relative value as novelty (42). This normalizes center signal against the spatial context information derived from the surround. Such function is equivalent to a comparison of spatially distinct areas of the image. The principle of antagonistic receptive fields is applied to all operators working on the image.

A conceptually similar problem arises in the temporal domain. Most of the objects in the real world are rigid and move with constant velocity (43). Information about them is contained in temporal discontinuities, which must be detectable regardless of ambient motion levels. Again, the limited response range of each operator necessitates continuous adjustment of operating characteristics to ambient local velocity. Hence, the system must normalize the temporal scale by resetting thresholds computed from relative, rather than absolute, values. Temporal information can be derived by comparing activities of two C/S operators of opposite polarity (3, 17). The time difference in the response waveshape of the two operators will produce a transient that carries the information about the onset/offset of change. This transient resembles a time derivative of intensity and is used to normalize the temporal scale.

It is clear from these spatial-temporal considerations that our visual system must first normalize intensity changes in space and time. And the way to subtract the DC component is to use antagonistic receptive fields implemented by lateral inhibition . The net result is a double representation of the image; one carrying spatial, and the other temporal, information. Both of them resemble the effect of convolving the original visual information with a center-surround filter resembling difference of Gaussians (DOG) (2, 3).

1b. Structural details

Our computing architecture for normalization of the lightness scale was inspired by natural vision systems (41). Major structural components include lateral interactions between neighboring elements within a layer, and converging and diverging pathways between the layers (fig. 4). Overlap between operators helps to enrich representation of the contrast information across the boundaries between different receptors. For the sake of simplicity local structures remain constant across the module.



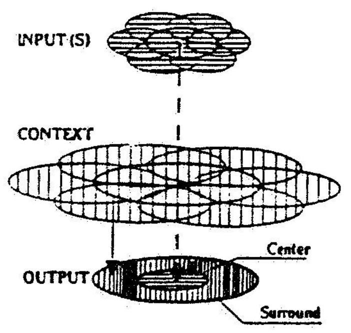


Figure 4. Flow of information in the generalized, normalization module (a). Center-surround antagonism of an output operator. Seven large context operators determine the surround response and seven smaller input operators determine the center response (b). The output operator compares the two responses.

The input to the spatial module is analogous to a layer of cone photoreceptors arranged in hexagonal array. The output operators, functionally resemble bipolar cells found in the vertebrate retina. Their responses are normalized by subtracting a local average computed by context operators. There are two types of output operators which differ in polarity and time to peak of response. The context information is always of opposite polarity to the center signal. Representing all relative values in the form of two opposite polarity masks improves stability of spatial-temporal interpolation and provides phase-like information about the original signal (45). Also, positive and negative operators differ in their coverage of the visual field and hence in spatial information. Therefore, if both positive and negative operators display a zero-crossing or a peak, it is more likely that the phenomenon is not an artifact created by noise, but a sign of significant discontinuity in intensity.

The comparison performed by output operators combines two levels of resolution in the sense that the large RF operators set the thresholds for the small ones in their area of activity. Other approaches are possible (46, 47), but for our initial implementation, we selected the simplest solution. The result computed at the output is then roughly the averaged second difference of the input intensities. Our method of combining the different levels of resolution is of particular interest because it was implemented using a simple, universal architecture based upon lateral inhibition. To simplify the simulation, we assume that the photoreceptors converging onto a given surround or output layer operator are linearly combined and that inhibition is a simple linear operation.

Fig. 5. shows combined architecture of both modules. Modularity and parallelism simplifies signal processing without any ad-hoc assumptions about image statistics. The temporal module also consists of three, functionally distinct, two-dimensional layers of C/S operators, arranged in a regular hexagonal lattice. The centers of the RF's overlap, and their sizes are different in distinct layers. To facilitate simulation, we chose to model only a small part of the visual field; hence we may assume that the sizes of the RF's remain constant throughout each layer.

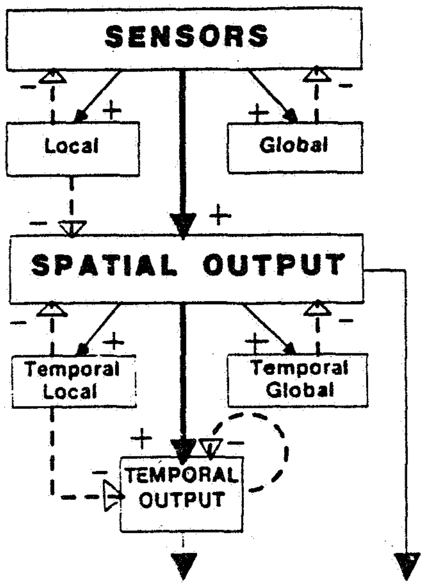


Figure 5. Hierarchical architecture of integrated spatial and temporal modules.

The input to the temporal module are two signals (I+ and I-) generated by the spatial module. They are of opposite polarity, display differences in their temporal behavior, and are regularly interspaced. Half of the temporal input operators receive I+ and the rest I-. A spatial discontinuity appearing at time t will generate a maximal response to I+ at t1 and I- at t2 with t1 and t2 not equal. This difference carries the information about the onset of temporal changes.

The time derivative is computed by an input operator which compares the information about the present input signal with values in the recent past. The source of the information about past values is feedback from the context operators. The feedback from global and local temporal context operators does not interfere with a signal normalized in space by the first submodule. This is similar to the action of local synaptic effect in amacrine cells. Context operators act to predict the future transient response to motion. The normalization of the temporal scale is achieved by shifting the velocity-response curve of the output operator over the domain of target velocities. This is based on comparison of motion in spatially separated areas of the visual field. Conceptually, DOG of dI/dt computes temporal information which appears as a transient. However, with rapid motion, when input intensity in the center of RF fluctuates sharply, large positive and large negative derivatives could cancel during the computation of the Gaussian. Hence, the need for the DOG of the absolute value of the derivative (48). The context layer operators, which compute the feedback, rectify in the sense that negative signals are attenuated. In biological systems, the amacrine and ganglion cells rectify in order to facilitate frequency coding used in the transmission of signals over long distances (3). This rectification is functionally similar to taking absolute values in our module.

1c. Simulation results

Fig. 6. is a simple demonstration of the lightness constancy function. The output signal from the module does not change significantly as the uniform background intensity is varied. Here input, represented by the horizontal axis, is *intensity on a logarithmic scale with 0 log units being* equivalent to darkness. The vertical axis represents the logarithm of the difference between the extreme responses on the light and dark sides of the discontinuities.

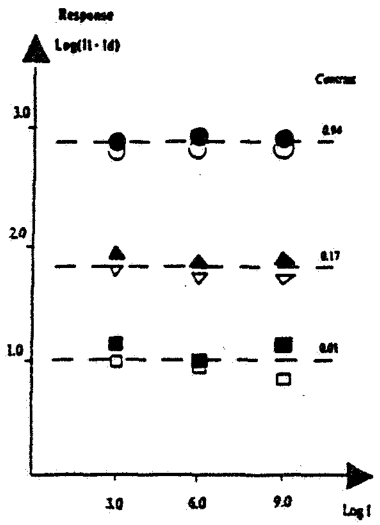


Figure 6. Response of the spatial module to increasing contrast at various levels of ambient light intensity.

The behavior of our module in response to a moving discontinuity of intensity is shown in Fig. 7. where the vertical axis represents the maximal response of an output operator in the center of the visual field. The horizontal axis represents velocity in interoperator units per iteration. One interoperator unit is the distance between two neighboring input operators. One iteration is the amount of time it takes for a signal to go from an input operator to the context layer and back to the same input operator via the immediate feedback. In all cases the input signal is a sharp discontinuity of intensity. The part of the discontinuity in the center of the visual field is moving at one velocity and the rest of it is moving at a possibly different velocity. These are called local velocity (vl) and global velocity (vg) respectively. Fig. 7. shows that if velocity is constant throughout the visual field, the response is small and almost independent of vl. However, if motion is restricted to a small part of the visual field (i.e. vg = 0) , a roughly linear response is obtained. This illustrates the fact that our module detects relative rather than absolute motion.

2. A neural net to extract motion heuristics.

The internal representation of the world that is used by a visually guided robot must be updated and maintained using the sensory data derived from the environment. Establishing a correspondence between the viewer-centered sensor data and an object-centered internal representation is an expensive computational task (49). Therefore, a roving robot must either sit for a while and contemplate it's new position, or move under assumptions which are a few steps behind the real world (50). Typically, the correspondence process forms an initial match between a perceived object and its internal model and then, as the object moves with respect to the roving robot, the orientation of the model may need to be updated to reflect current sensor information (51). This paper demonstrates how a connectionist architecture can speedup the matching of an internal 3-D model to changing edge features, by precomputing future positions of the edge features and providing the matcher with heuristic information describing in which direction to start manipulating the model.

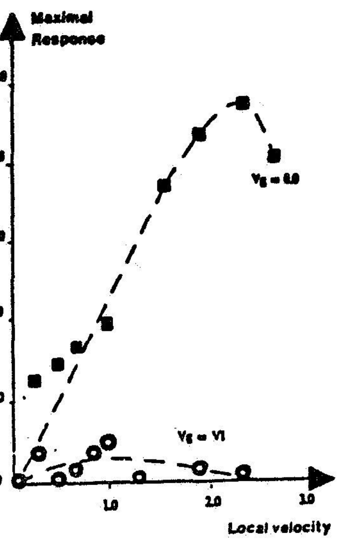
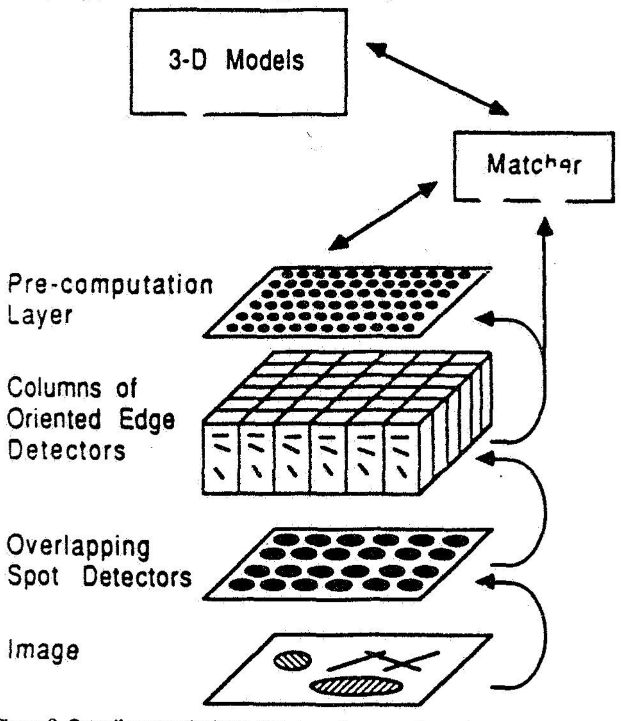


Figure 7. Motion throughout the visual field produces a small response (Vg=Vl) Motion in a small region induces a large roughly linear response (Vglob=0).

The recognition of an object must involve the matching of some input data to an internal representation of an object. The matching can be accomplished by either: 1) manipulating the data and comparing it to a set of fixed models or, 2) transforming the model to match the captured edge features. As an object in a scene moves, the 2-D projection of its boundaries and key features appear to undergo translation, rotation, and occlusion. This suggests that the second method is more natural because we do not need to compute the position of occluded edges. Also, the second method is more suitable for a goal-driven system (52). The constantly updated model becomes a representation of the world that can support scene interpretation, planning, and other higher-level cognitive functions. Manipulating the model requires the matcher to rotate and translate its internal model in an attempt to match the current edge features. In this approach the internal model is continuously trying to catch up to the real world. A speed-up would occur if the matcher received, along with the incoming data, a preliminary guess of which way the features were rotating or translating.



3-D Models
Matcher
Pre-computation Layer
Columns of Oriented Edge Detectors
Overlapping Spot Detectors
Image

Figure 8. Overall connectionist architecture of a network used to extract motion heuristics.

Most existing matchers are based on graph theoretic algorithms which execute in exponential time with respect to complexity of the graph description (53). The matcher establishes a correspondence between the internal model representation and the edge features of an image. In this paper, we assume that this part of the matcher is given. We are concentrating on the problem of how the matcher can maintain the established correspondence as an object is undergoing smooth or discontinuous motion.

To maintain the correspondence, the matcher could precompute numerous, new orientations of the internal model and have them ready for incoming data. But this precomputation technique would be time consuming and unwieldy, since it substantially increases the graph size. Incoming data, though, can be used to give specific suggestions on how the matcher should manipulate a model. A technique for precomputing possible future positions of the edge features is the first step in formulating a model manipulation heuristic for the matcher.

By using a connectionist architecture (9), we hope to understand how visual functions can be derived from massively parallel computing structures. Additionally, neurophysiological evidence can be used to inspire possible interconnectivity solutions (17, 54,55). Our mechanism for precomputation is partially motivated by the structure of the early visual cortex which has been extensively reviewed elsewhere (56). This region of the cortex is composed of vertical slabs which contain neurons sensitive to contrast edges, of a preset orientation, that are in particular region of the visual field (56, 57). Within each slab, there is also a convergence of information related to color and motion.

We have limited our implementation of vertical slabs to the simulation of their edge orientation information. In a fashion similar to the visual cortex, edge detectors of differing orientations over the same spatial subregion are grouped together and locally interconnected. Such a group of oriented edge detectors are called a column. A column contains all of the available orientation information for its particular sub-region of the image. In the future, we hope to more realistically model the robustness of the vertical slabs in the visual cortex.

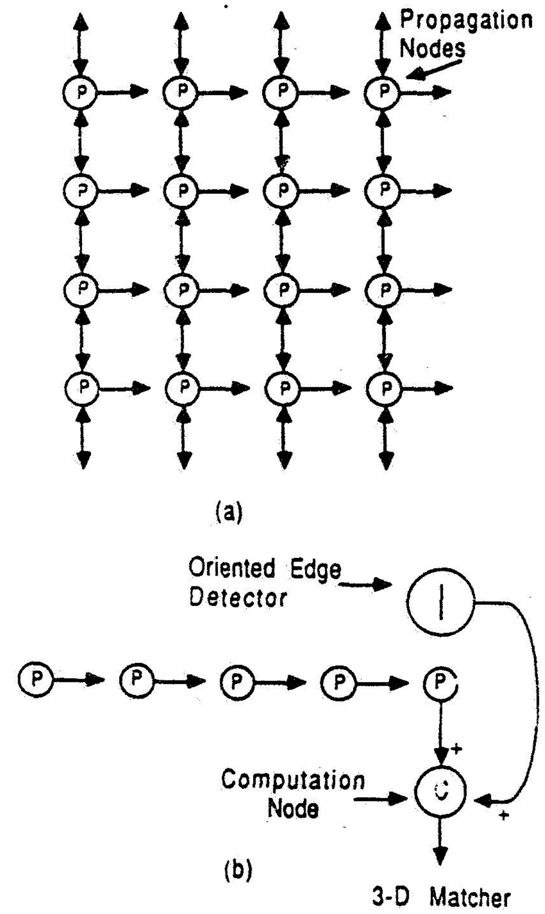


Figure 9. Propagation nodes are interconnected to propagate the direction-specific activities of edge detectors (a). A computation node requires simultaneous activities in both, its propagation node and its edge detector, before it will signal the response.

Fig. 8 outlines the computational hierarchy of the architecture. The light from a scene is initially transduced into electrical signals by a layer of photosensors. These signals are then processed by spot detectors which are sensitive to local changes in image intensity. The center receptive fields of the spot detectors overlap each other by thirty percent. The output from the spot detectors are grouped to form oriented edge detectors which are then organized into columns. It should be noted that this implementation deliberately differs from the known neurophysiological data because of the limitations of our simulation tools. Surrounding each edge detector are propagation nodes which compute where the edge may move in the future by exciting the propagation nodes in adjacent columns. Oriented edge information is used by both the precomputation layer and the matcher. The precomputation layer gives the matcher heuristic information on the direction of a moving edge feature. Computation nodes which are in this layer are able to guess at the direction of an edge by comparing the excitation levels of oriented edge operators and the surrounding propagation nodes.

The lowest layer of the architecture extracts changes in image intensity by using center-surround receptive fields has been detailed in (18). Briefly, the image is first filtered by a layer of nodes with center/surround antagonistic receptive fields. To reduce the simulation complexity, this layer was modeled using a convolution operator.

The analyses of the information available from motion makes it apparent that there are only few possible directions that edge feature could take without violating the heuristics used for matching points in separate images (58). Considering only rigid physical objects with limited velocity, the motion is limited to a few possible next-frame positions and directions. Hence in principle it is possible to simultaneously tell the matcher where the edges are and how they are moving.

To accomplish this objective, we organize the oriented edge detectors within a sub-region of the image into a column and then bring the columns together to form a cube. A transverse slice of the cube contains all of the edge detectors of a particular orientation over the entire image. When an edge becomes active, indicating that the current image has an edge feature at that location and orientation, we want to use that fact to prepare for future movement of that edge feature.

A moving edge feature can at most activate one of six, nearest-neighbor edge detectors in our hypercolumn. To monitor this change, each oriented edge detector in a column is connected to six propagation nodes (p-nodes), four translational and two rotational. Thus, a specific p-node will transmit the activity of its edge detector in one of the six possible directions. By propagating the excitation of an edge detector, the p-nodes prime the network for specific, future orientations of an edge feature (fig. 9).

A computation node (c-node) combines the information from an oriented edge detector and its associated p-nodes. A c-node will only fire when its edge detector and one p-node are high. Of course prior to arrival of the edge feature, high activity of one p-node implies potential direction of motion that can be signaled to the matcher.

2a. Example

Fig. 10 illustrates the changing excitation levels of the p-nodes and c-nodes over time. In this example, a bar is moving from left to right across the visual field (Fig. 10a) Fig. 10b demonstrates how the excitation levels of edge detectors are being propagated, in a rightward direction, by the +y translational p-nodes. When both the p-nodes and the edge detectors are excited, the c-nodes will momentarily fire (fig. 10c) and provide heuristic information to the matcher.

The precomputation layer of our connectionist architecture can provide heuristic information useful in matching 3-D models to time-varying edge features. If the velocity of an edge feature should exceed the propagation rate of the p-nodes, then the c-nodes will not be excited and the matcher will not receive any heuristic information. The matcher could interpret such an edge as being part of either, a new object in the scene or, an object that is undergoing discontinuous jumps.

CONCLUSION

New approaches to machine sensing and perception were presented. The motivation for crossdisciplinary studies of perception in terms of AI and Neurosciences is suggested. The question of computing architecture granularity as related to global/local computation underlying perceptual function is considered and examples of two environments are given. Finally, the examples of using one of the environments, UCLA PUNNS, to study neural architectures for visual function are presented.

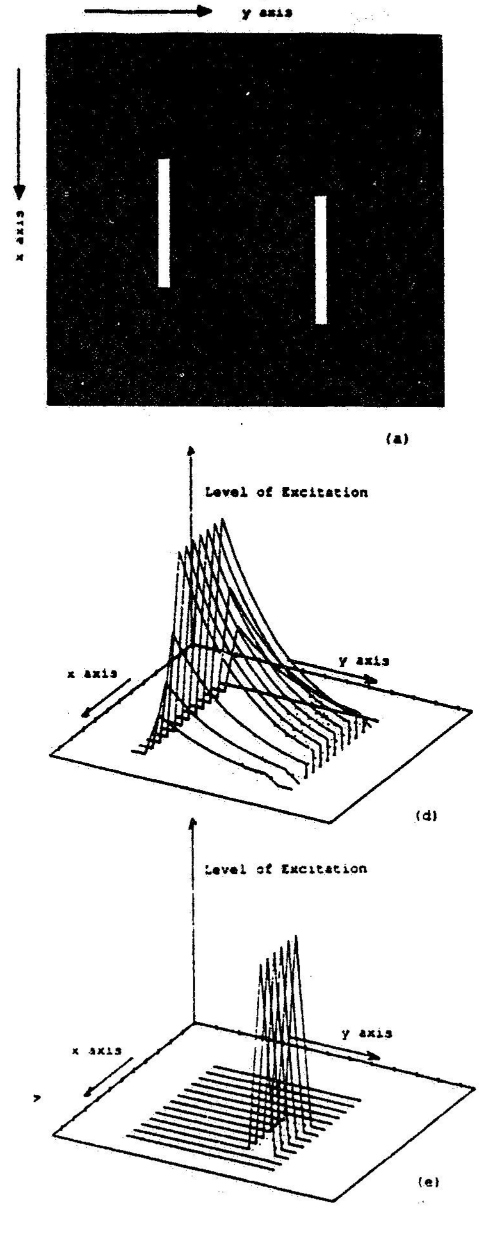


Figure 10. The stimulus is a time-varying image of the vertical bar is moving from left to right (a). The P-nodes propagate the exponentially decaying signal about vertically oriented edge moving in the +Y direction (b). When the bar moves to the right, the C-node becomes active and sends the information to the matcher that this edge feature has undergone left to right translation

REFERENCES

1. Barr, A., & Feigenbaum, E., The handbook of artificial intelligence. 1982, Heuristech Press.
2. Marr, D., Vision. 1982, Freeman and Co.
3. Rodieck, R.W., *The Vertebrate Retina*, W.H. Freeman and Co., San Francisco, 1973.
4. Barlow, H.B., Why have multiple cortical areas? Vis. Res. 26 pp.81-90, 1986.
5. MacLeod.R.B & H.L.PickJr., Perception. 1974 Cornell University Press.
6. Schank, R., & Abelson, R., Scripts, Plans, Goals and
7. Rock.I., The logic of perception, 1983, MIT Press.
8. Henderson T.C., Wu So Fai, and Hansen C., "MKS: A Multisensor Kernel System," IEEE Trans. Sys., Man, Cybern., vol. SMC-14, no. 5, pp. 784-791 (1984);
9. Feldman, J.A., "Connectionist Models and Parallelism in High Level Vision", *Computer Vision, Graphics, and Image Processing*, 31, pp. 178-200, Feb. 1985.
10. Hinton,H., & J.Anderson, Parallel models of associative memory, 1981 L. Earlbaum Associates Inc.
11. Rosenblatt, F., *Principles of Neurodynamics*, Spartan Books, Washingtion D.C., 1962.
12. Feldman, J.A., and Ballard, D.H., "Connectionist Models and Their Properties", *Cognitive Science*, 6, pp. 205-254, 1982.
13. Hopfield, J.J., "Neural networks and physical systems with emergent collective computational properties." Proc. Natl. Acad. of USA 1982, 79, 2554-2558.
14. Ballard, D.H., "Parmeter Networks: Towards a Theory of Low Level Vision", TR 75, Computer Science Dept., University of Rochester, April 1981.
15. Hopfield, J.J. & Tank, D.W., "Neural computationa in constraint satisfaction problems and the traveling salesman." Biol. Cyber, in press.
16. Winston P.H., "Artificial Intelligence," Addison-Wesley, Reading, Mass. (1984).
17. Skrzypek, J., "A unified computational architecture for preprocessing visual information in space and time." ANRT/SPIE Proc of Intl. Symp. on Computer Vision and Intelligent Robots, Canes, France 1985.
18. Gungner, D., and Skrzypek, J., Connectionist Architecture for Matching 3D Models to Moving Edge Features, *SPIE Conference on Intelligent Robots and Computer Vision*, Vol. 726, October, 1986.
19. Kuffler, S.W., and Nichols, J.G., *From Neuron to Brain*, Sinauer Associates, Inc. Massachusetts, 1976.
20. Buchanan,B.G., & Shortliffe, E.H.,(eds) Rule-based Expert Systems: The mycin experiment of the Stanford Heuristic Programming Project, Addison-Wesley, 1984.
21. Brady,M., & Paul,R., eds. Robotics Research, 1984, MIT Press.
22. Feldman, J., et al., "The Stanford artificial intelligence project", Proc. IJCAI, Washington D.C., 1969
23. Stiber, M., & Skrzypek, J., "LISP based PC Vision Workstation." SPIE Conf. Image Understanding and Man Machine Interfaces. Los Angeles. 1987
24. Michalski, R.S., Carbonell, J.G., & Mithcell, T.M., (eds.) Machine learning: An Artificial Intelligence Approach, Tioga publishing Co. Palo Alto, 1983.
25. Thagard, P., Parallel Computation and the Mind-Body Problem, *Cognitive Science*, 10, 301-318, 1986.
26. Toffoli, T., Physics and Computation, *International Journal of Theoretical Physics*, Vol. 21, Nos. 3/4, 1982.
27. Von Neumann, J., Second lecture of Theory and Organization of Complicated Automata, in *Theory of Self-Reproducing Automata*, Burks, A.W. ed., University of Illinois Press, 1966.
28. McCulloch, W.S., and Pitts,W., A Logical Calculus of the Ideas Immanent in Nervous Activity, *Bulletin of Mathematical Biophysics*, Vol. 5, 1943.
29. Hebb, D.O., *The Organization of Behavior*, John Wiley, New York, 1949.
30. Rosenblatt, F., *Principles of Neurodynamics*, Spartan Books, Washingtion D.C., 1962.
31. Rumelhart, D.E., McClelland, J.L., et.al., *Parallel Distributed Processing*, MIT Press, 1986.
32. Perkel, D.H., A Computer Program for Simulating a Network of Interacting Neurons, *Computers and Biomedical Research*, Vol. 9, pp. 31-43, 1976.
33. Wittie, L.D., Large-scale Simulation of Brain Cortices, *Simulation*, Vol. 31, Num. 3, September, 1978.
34. Small, S.L., Shastri, L., Brucks, M.L., Kaufman, S.G., Cottrell, G.W., Addanki, S., ISCON: A Network Construction Aid and Simulator for Connectionist Models, *University of Rochester, Department of Computer Science Technical Report*, TR 109, April, 1983.
35. Fanty, M., Goddard, N., User's Manual Rochester Connectionist Simulator - Draft, *University of Rochester, Department of Computer Science Technical Report*, September, 1986.
36. Mesrobian, E., and Skrzypek, J., Connectionist Architecture for Computing Textural Segmentation, *SPIE Conference on Image Understanding and the Man-Machine Interface*, Vol. 758, January, 1987.
37. Hubel, D.H., and Wiesel, T.N., Brain Mechanisms of Vision, *The Mind's Eye*, W.H. Freeman and Company, New York, 1986.
38. T. Binford., "Inferring Surface Information from Images", Artificial Intelligence volume 17, pp. 205 -241, 1981.
39. E.Land. and J.J.McCann., "Lightness Theory", Journal of the Optical Society, volume 61, number 1, January 1971, pp. 1 -11.
40. J. Skrzypek and D. Shulman, "Preprocessing Using Multiresolution Lateral Inhibition", SPIE Proc. Cambridge Conf. on Intelligent Robots and Computer Vision, Cambridge 1984, 91-97.
41. J, Skrzypek. and F, S. Werblin., "Lateral Interactions in the Absence of Feedback to Cones", Journal of Neurophysiology volume 49, pp.1007-1016. 1983.
42. F, Ratliff., Mach Bands: Quantitative Studies on Neural Networks in Retina, Holden Day 1965.
43. E. C. Hildreth, "The Measurement of Visual Motion", Ph.D. Thesis M.I.T., Department of Electrical Engineering. 1983.
44. K, Naka. and Rushton. W.A.H., "S Potentials from Intensity Units in the Retina of Fish (Cyprindidae)", Journal of Physiology, volume 185, pp. 587-599. 1966.
45. S. W. Zucker and R. A. Hummel, "Receptive Fields and The Representation of Visual Information", IEEE Conference On Pattern Recognition, Montreal 1984, 515-517.
46. J.F. Canny, "Finding Edges and Lines in Images", M.I.T. Artificial Intelligence Laboratory Technical Report 720. 1983.
47. D. Terzopoulos, "Multilevel Reconstruction of Visual Surfaces: Variational Principle and Finite Element Representations " in Rosenfeld, A., (ed.) Multiresolution Image Processing and Analysis, Springer Verlag, 1983.
48. D.Shulman and J.Skrzypek, "Multiresolution temporal preprocessing and lateral inhibition.", SPIE proc. Cambridge Conf on Intelligent Robots and Computer Vision, Cambridge 1985.
49. Shepard, R.N., and Cooper, L.A., *Mental Images and Their Transformations*, MIT Press, 1986.
50. Lowrie, J.W., Thomas, M., Gremban, K., and Turk, M., The Autonomous Land Vehicle (ALV) Preliminary Road-following Demonstration, *Intelligent Robots and Computer Vision*, SPIE Vol. 579, pp. 336-350, Cambridge, Massachusetts, 1985.
51. Tsuji, S., Osada, M., and Yachida, M., Tracking and Segmentation of Moving Objects in Dynamic Line Images, *IEEE Pattern Analysis and Machine Intelligence*, Vol. PAMI-2, No. 6, November 1980.
52. Pinker, S., Visual Cognition: An Introduction, in *Visual Cognition*, Pinker, S. ed., pp. 1-63, MIT Press, 1985.
53. Ballard, D.H., and Brown, C.M., *Computer Vision*, Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1982.
54. Marr, D., and Poggio, T., A Theory of Human Stereo Vision, *Proc. Roy. Soc. London B*, Vol. 204, pp. 301-328, 1979.
55. Ballard, D.H., Parameter Nets, *Artificial Intelligence*, Vol 22, pp. 235-267, 1984.
56. Hubel, D.H., Evolution of Ideas on the Primary Visual Cortex, 1955-1978: A Biased Historical Account, Nobel Lecture, December 8, 1981, *Les Prix Nobel*, Stockholm, Sweden, 1981.
57. Marr, D., and Hildreth, E., Theory of Edge Detection, *Proc. R. Soc. London B*, Vol. 207, pp. 187-217, 1980.
58. Prager, J.M., Segmentation of Static and Dynamic Scenes, *COINS Technical Report*, 79-7, Computer and Information Science, University of Massachusetts, May 1979.
59. Gungner, D., & Skrzypek, J. "UCLA PUNNS - A neural network machine for computer vision." SPIE Conf. Image Understanding adn Man-Machine Interfaces., Los Angeles, 1987.

S19-62
78
188139

Parallel Processing for Digital Picture Comparison

H.D. Cheng and L.T. Kou
University of California, Davis
Davis, CA 95616

CC907079

ABSTRACT

In picture processing an important problem is to identify two digital pictures of the same scene taken under different lighting conditions. This kind of problem can be found in remote sensing, satellite signal processing and the related areas. The identification can be done by transforming the gray levels so that the gray level histograms of the two pictures are closely matched. The transformation problem can be solved by using the "packing" method. ~~In this paper,~~ we propose a VLSI architecture consisting of m x n processing elements with extensive parallel and pipelining computation capabilities to speed up the transformation with the time complexity O(max(m,n)), where m and n are the numbers of the gray levels of the input picture and the reference picture respectively. If using uniprocessor and a dynamic programming algorithm, the time complexity will be O(m^3xn). The algorithm partition problem, as an important issue in VLSI design, is discussed. Verification of the proposed architecture is also given.

Index terms: Digital picture comparison, packing algorithm, very large scale integration (VLSI), algorithm partition, VLSI architecture verification.

I. INTRODUCTION

The technique of dynamic programming has wide applications in computer science [6,7] for solving mathematical problems arising from multistage decision processes. Based on the dynamic programming path-finding algorithm, the technique of dynamic programming is both mathematically sound and computationally efficient. The recent advent of very-large-scale integration (VLSI) technology has triggered the thought of implementing some algorithms directly in hardware with extensive parallel and pipelining computation capabilities. The use of VLSI architectures to implement dynamic programming procedures has been investigated for several applications. Guibas et al. [8] describes a VLSI architecture for a class of dynamic programming problems characterized by optimal parenthesization. Chu and Fu [9] describe VLSI architectures for recognition of context-free and finite-state languages. Chiang and Fu [10] describe a VLSI implementation of Early's algorithm for parsing general context-free languages. Cheng and Fu [11] describe algorithm partition and parallel recognition of general context-free languages using fixed-size VLSI architecture. Liu and Fu [12] describe a VLSI implementation for string-distance computation. Clarke and Dyer [15] describe four VLSI architectures for line and curve detection. Cheng and Fu [13,14] propose VLSI architectures for pattern-matching and hand-written symbol recognition. In this paper, we propose a VLSI architecture for identifying digital pictures if they are taken from the same scene under different lighting conditions. This is a very important problem related to remote sensing, satellite signal processing and other areas. As an important issue in VLSI design, the algorithm partition problem is discussed. The backtracking procedure is also discussed in much detail, and the formal verification of the proposed architecture is given. An example is used to illustrate the work of the proposed VLSI architecture.

II. PRELIMINARY

The image matching technique has been used extensively for many applications such as curvature sequences detection [2], template matching and pattern matching [1], character recognition, target recognition, aerial navigation and stereo mapping, picture matching, earth resource analysis, missile guidance, intelligence gathering systems, and robotics [2,3].

There are many situations in which we want to match or register two pictures with one another, or match some given pattern with a picture [2]. For example:

(a) Given two or more pictures of the same scene taken by different sensors, we want to determine the characteristics of each pixel with respect to all of the sensors and then we can classify the pixels.

(b) Given two pictures of scenes taken from different times, we want to determine the points at which they differ and then can analyze the changes that have taken place.

(c) Given two pictures of a scene taken from different positions, we want to identify corresponding points in the pictures and then determine their distances from the camera to obtain three-dimensional information from the scene.

(d) We want to find places in a picture where it matches a given pattern.

In this paper we want to discuss another very important aspect in picture processing which is to identify two digital pictures of the scene taken under different lighting conditions. These kinds of problems arise from many areas such as remote sensing, satellite signal processing, and etc. The identification can be done by transforming the gray levels so that the gray level histograms of the two pictures are closely matched.

Mathematically, a picture is defined by a function of two variables $F(x,y)$, where $F(x,y)$ is the brightness, or K-tuples of brightness values in several spectral bands, [2,4,5] and x and Y are the coordinates in the image plane. In the black and white case, the values are called gray levels. These values are real, non-negative, and bounded. The pictures are represented as matrices with integer elements which are the pixels. A gray level histogram of an image is a function that gives the frequency of occurrence of each gray level in the image. Where the gray levels are quantized from 0 to n, the value of the histogram at a particular gray level p, denoted $H(p)$, is the number of fraction of pixels in the image with that gray level [5]. When pictures of the same scene are obtained under different lighting conditions, different histograms are gained. For identifying these pictures, we can transform their gray level scales so that their histograms would closely match each other.

Assume that H_1 and H_2 are histograms of two pictures obtained from the same scene with m and n gray levels, respectively. An algorithm is proposed to "reshape" H_1 (i.e. rescale its gray levels) so that it has the minimal deviation from H_2. The mathematical problem is defined by:

$$Z = \min_{(X_0,\dots,X_n)} \sum_{j=1}^{n} |H_2(j) - \sum_{i=P}^{Q} H_1(i)|$$

where $P = X_{j-1}$ and $Q = X_j - 1$ subject to

$$1 = X_0 < X_1 < \dots < X_n = m+1 \qquad (1)$$

X_i = integer, for $i = 1,\dots,n$.

It will transform the gray levels $X_{j-1},\dots,X_j-1$ in one picture into gray level j in the other picture, for suitably chosen X_{j-1} and X_j, $j = 1,\dots,n$.

This problem can be interpreted as a packing problem: to pack m objects of sizes $\{H_1(1),\dots,H_1(m)\}$ into n boxes of spaces $\{H_2(1),\dots,H_2(n)\}$ in such a way that

(i) if the i^{th} object has been placed in the j^{th} box, the $(i+1)^{th}$ object is not allowed to be packed into the k^{th} box for any $K < j$, and

(ii) the accumulated error due to space over-packed of leftover is minimized.

Such a problem can be solved by using dynamic programming techniques. Let $S_j(i)$ be the minimal accumulated error caused by transforming the gray levels $1,\dots,i$ into the gray levels $1,\dots,j$. The recursive formula is given by

$$S_j(i) = \min_{0<u<i} \{S_{j-1}(u) + |H_2(j) - \sum_{v=u+1}^{i} H_1(v)|\} \qquad (2)$$

for $i=1,\dots,m$ and $j=1,\dots,n$

where the initial conditions are $S_0(0) = 0$, $S_0(i) = \sum_{v=1}^{i} H_1(v)$ for all $i = 1,\dots,m$ and $S_j(0) = \sum_{u=1}^{j} H_2(u)$ for all $j = 1,\dots,n$. If $i>j$, then $\sum_{k=i}^{j} H_1(k) = 0$. The minimal accumulated error, $S_n(m)$, can be computed.

The straight forward execution of this procedure would obtain the optimal solutions for all (i,j) pairs with time complexity $O(m^3 x n)$ by using uniprocessor. In this paper, we want to propose a m x n VLSI array to speed up the computation. The time complexity for the proposed architecture is $O(\max(m,n))$.

III. VLSI DIGITAL PICTURE COMPARATOR

3.1 The algorithm and its VLSI implementation

We will propose a VLSI architecture based on the space-time domain expansion approach [14,15], which has a very natural and regular configuration and can be implemented easily by applying today's VLSI technology. Another important issue in VLSI design - algorithm partition problem is also solved by using the proposed VLSI architecture. The proposed VLSI architecture can speed up the digital picture comparison procedure greatly by using extensive parallel and pipelining techniques. Before discussing the VLSI architecture in detail, we propose the following algorithm.

Let H_1 and H_2 be the histograms of two pictures taken at the same scene with m and n gray levels, respectively.

Algorithm 1: The algorithm for digital picture comparison:

```
begin
    $S_0(0)$ := 0;
    for i := 1 to m do
        begin
          $S_0(i)$ := 0;
          for k := 1 to i do
          $S_0(i)$ := $S_0(i)+H_1(k)$
        end;

    for j := 1 to n do
        begin
          $S_j(0)$ := 0;
          for k := 1 to j do
          $S_j(0)$ := $S_j(0)+H_2(k)$
        end;

    for i := 1 to m do
      for u := 0 to i-1 do
        begin
          v := u + 1;
          T(v) := 0;
          for k := v to i do
          T(v) := $H_1(k)$ + T(v)
        end;

    for i := 1 to m do
      for j := 1 to n do
        begin
          T' := $S_{j-1}(i)$ + $H_2(j)$;
          I := i (index channel value);
          T := 0;
          for u := 1 to i-1 do
            begin
              v := u +1;
              s := 0;
              for k := 1 to v do
                begin
                  s := s + $H_1(k)$;
                  T := $|H_2(j) - s|$;
            T := $s_{j-1}(u)$ + T;
            If $T'$ < T then
            $t'$ := T and output v to the index channel by letting I := v
          end
        end;
    end
```

Append index-pair (I,j-1) to index-pair (i,j), when the identification signal arrives, and form ((I,j-1), (i,j)).

end.

We can build a VLSI array with m x n processing elements. Each processing element has a subtracter which will produce the absolute value of the two inputs difference, a comparator which will compare two input values and output the smaller value with the corresponding index to the next processing element below it. The functions performed by the (i,j)th processing element are as follows:

Input: $H_2(j)$, outputs of (i-1,j)th processing element, $\sum_{v=1}^{i} H_1(v)$, index-pair, $S_{j-1}(i-1)$ and $S_{j-1}(i)$.

Output: $S_j(i)$ and index-pair to the right element when the identification signal arrives, and the intermediate results to the processing element below.

Operations: Each processing element has a local connection to the processing element beneath it which will accept the intermediate results including the accumulated errors and the index-pairs, and has a local connection to the right processing element which will receive $S_j(i)$ and index pair (i,j) when the identification signal arrives. Each processing element can perform accumulation, $|a-b|$, and comparison operations, and requires one time unit. The adder uses the combinational circuit, which will not require the time unit, or its delay is much smaller than a time unit. The data will move among the processing elements, one processing element per time unit.

1) Input data of H_1 arrives at the (i,j)th PE and performs the accumulation of each for one time unit. $|H_2(j)-\sum_{v}^{i}H_1(v)|$ needs one time unit.

2) $S_{j-1}(i-1)$ arrives at the (i,j)th processing element and it performs $S_{j-1}(i-1) + |H_2(j)-H_1(i)|$ operation which requires one time unit. The result is delayed one time unit.

3) $0<u<i-2\{S_{j-1}(u)+|H_2(j)-\sum_{v=u+1}^{i} H_1(v)|\}$ arrives at the (i,j)th processing element from the (i-1,j)th processing element and compares with the result of step 2) which requires one time unit. At the same time, the identification signal arrives and the result will compare with $S_{j-1}(i)+H_2(j)$ which will require one time unit. Then $S_j(i)$ and the index-pair will be sent to the (i,j+1)th processing element.

Algorithm 2 VLSI implementation of Algorithm 1

Input: Gray levels of the input picture $-H_1(i)$, and of the reference picture $-H_2(j)$ (for $1<i<m$, $1<j<n$); indices, index pairs; initial conditions: $S_0(0)$, $S_0(i)$, and $S_j(0)$ (for $1<i<m$, $1<j<n$); and identification signals.

Output: The accumulated error $S_j(i)$ and corresponding index pairs

Move the gray levels $H_2(j)$ of the reference picture, the identification signals, and the index j from the top to the bottom one processing element per time unit. Move the gray levels $H_1(i)$ of the input picture and index i from the left to the right of the VLSI array one processing element per time unit. The identification signals will be sent at the fifth time unit and will move down one processing element per two time units and move to the right one processing element per time unit. When the identification signal arrives, it will open the connection channel to the comparator which connects the right processing element, and $S_j(i)$ will be sent to the processing element (i,j+1). To obtain the 'packing' sequence, we have to perform a backtracking procedure which can be done in several ways as follows.

1) Output the accumulate error matrix S and/or the index-pairs to the host machine which will perform the backtracking procedure.

2) Attach another VLSI module and use the tag of the index pair as the search key to perform the backtracking procedure.

3) Expand the 'append' operation to the one which appends the index into the index list of its ancestor.

An index list is formed by appending an index or an index list. We can use index (m,n) as the tag to find the 'packing' sequence. This will change the backtracking procedure into forward and speed up the computation, but it requires a large output channel capacity, especially for the processing elements located at the upper-right corner. The upper bound of the channel capacity for the (i,j) processing element will be (i+j+1).

4) Add an index register to each processing element which consists of two parts, the first part for the first index and the second part for the second index. The second part of the index pair register will compare with the tag. If they are matched, the second part is output into the output channel and also output into the first part as the tag to its top and left side neighbors. The tag will move up until it match with another index pair. The procedure will be continued. We need one index register for each processing element. At the (2i+j+3)th time unit, send a backtracking signal which moves along the channels connecting to the left neighbor and the one on the top of it, each processing element per time unit. The index (m,n) is used for the tag of the (m,n)th processing element. It needs at most (m+n) time units to complete this procedure.

From the above discussion, we can conclude that the proposed architecture can compare two digital pictures by transforming the gray levels. In many applications, only the summation error is required. In such cases, we can simplify the structure of the processing element and the entire VLSI architecture further. If there are P digital pictures which are compared with the reference picture, or an input digital picture compared with P reference digital pictures, we can make a P-time expansion. The time complexity will be O(max(Pxm,n)). If using uniprocessor, the time complexity will be $O(Pxm^3xn)$. For indicating the most matched digital picture, we number the digital pictures and add a register consisting of two parts. One part is for the summation error, the other is for the index of the numbered digital pictures. We also add a counter which is initially set to zero and starts at (2m+n+3)rd time unit.

The operation of the register is as follows:

```
begin
 error.register:=∞;
  if error.register > error array
     then begin
           error.register:=error.array
           index.register:=counter
         end end
```

The final result of index.register indicates the index of the most matched digital picture. If we use a three dimensional array (Pxmxn processing elements), the time complexity will be reduced to O(max(P,m,n)). The detail will be omitted here.

IV. VERIFICATION OF THE PROPOSED ALGORITHM

To verify algorithm 2, we need the following lemmas and theorem.

Lemma 1: The identification signal arrives at the (i,j)th processing element at the (2i+j+2)th time unit.

Proof: The identification signal is sent at the fifth time unit and it needs 2(i-1) time units to reach the ith row, it then needs j time units to arrive at the (i,j)th processing element. Totally, 5+2i-2+j-1= 2i+j+2 time units.

Lemma 2: $\sum_{v}^{i} H_1(v)$ will be computed at the (v,j)th processing element at the (i+v+j-2)th time unit, for all $1 \le v \le i, 1 \le i \le m$ and $1 \le j \le n$.

Proof: First consider the j=1 case. From the data arrangement in Fig. 3, the first input of the vth row will arrive at the boundary of the array at 2(v-1)th time unit, then (i-v+1) time units are needed to compute $\sum_{v}^{i} H_1(v)$. Totally, 2(v-1)+(i-v+1)=i+v-1 time units are needed. Since the computation of the (v,k)th processing element will start one time unit earlier than one of the (v,k+1)th processing elements, the time units needed for the (v,j)th processing element will be i+v+j-2 to produce the summation.

Theorem: After receiving the inputs, $S_j(i)$ will be produced at the (2i+j+3)th time unit, for all $1 \le i \le m$ and $1 \le j \le n$.

Proof: We prove the theorem by induction on i and j.

Basis: First we consider i=j=1 case. Since $S_0(0)$ and $S_0(1)$ are fixed values which exists already, $H_1(1)$ the inputs into the processing element (1,1) and it performs the accumulation which requires one time unit.

Then $|H_2(1)-H_1(1)|$ is performed by spending another time unit. It will be added to $S_0(0)$ and delayed one time unit. At the forth time unit, the result will be compared with $S_1(0)$. When the identification signal arrives at the fifth time unit, the result of the comparator will compare with $S_0(1) + H_2(1)$ and output $S_1(1)$ which needs one more time unit, $6=2\times1+1+3=2\times i+j+3$.

Induction Step: Our induction hypothesis is that all (p,q)th processing elements can produce the outputs and the index-pairs at the $(2\times P+q+3)$th time unit, for all $1<p<i$ and $1<q<j$.

Now consider the $(i+1,j)$th processing element. According to lemma 2 and the hypothesis, $\sum_{v}^{i+1} H_1(v)$ will be computed at the (v,j)th processing element, $(i+1+v+j-2)$th time unit, for all $1<v<i$, $1<i<m$ and $1<j<n$ and the comparators are connected in a pipeline version, so $M = \min_{0<u<i} (S_{j-1}(u)+|H_2(j)-\sum_{v=u+1}^{i+1} H_1(v)|)$ will be output from the $(i,j-1)$th processing element, $(2i+j-1+2+3)$th time unit. Also $S_{j-1}(i+1)$ will be input at the $2\times(i+1)+j-1+3$th time unit. At the same time $N=S_{j-1}(i+1)+H_2(j)$ will be computed. According to lemma 1, the identification signal arrives at the $(i+1,j)$th processing element at the $2(i+1)+j+2$th time unit. Then M and N will be compared, the minimum $(M,N)=S_j(i+1)$ will be sent to the $(i+1,j)$th processing element at the $(2i+j+5)$th time unit. Since $S_{j+1}(i+1)$ will be one time unit later than $S_j(i+1)$, $S_{j+1}(i+1)$ will be obtained at the $(2i+j+6)= 2(i+1)+(j+1)+3$ the time unit. Therefore the proof is completed.

Corollary 1: The accumulated error and the index pairs can be obtained at the $(2m+n+3)$th time unit.

Proof: Follow the theorem and let $i=m$ and $j=n$.

V. ALGORITHM PARTITION

We could use a one-dimensional array or a two-dimensional array with size different to the problem size by performing time expansions following the partition rule.

A. Using the One-Dimensional Array

First we assume that the size of the array is m. We can consider it as an m-space expansion along the x_1 direction. The input channels will form the queues. The register will hold the initial value and the result from the CR output which will input into the register by the control signal. The control signal is sent at the $(m+1)$th time unit and moves down per two time units and one processing element. The input will repeat n times. The time complexity will be $O(m \times n)$.

B. Using the Two-Dimensional Array with the Dimensions Kxl

If $k=m$ and $\ell=n$, it is the case which has already been discussed. We now consider the other cases. According to the partition rule we have to make an $\lceil m/k \rceil$ - time expansion and an $\lceil n/l \rceil$ - time expansion. There are also queues for feedback of the data. The lengths of the queues will be varying with the values of m and n to make the right data meet at the right processing element at the right time. This will cause much difficulty to the control system and the queue structures. Hence, we either use a sufficiently large size VLSI architecture or use a one-dimensional array to solve the partition problem.

VI. CONCLUDING REMARKS

We have proposed an VLSI architecture for digital picture comparison. The time complexity will be $O(\max(m,n))$ by using a two-dimensional $m \times n$ array, where m is the gray level of the input digital picture and n is the ray level of the reference digital picture. With a uniprocessor, the comparison process will have the time complexity $O(m^3 \times n)$ if using the straightforward computation approach. If there are p reference pictures using the proposed architecture, the comparison process will be solved in time $O(\max(m \times p,n))$; and using a uniprocessor the time complexity will be $O(m \times p \times n)$. If using a three-dimensional array, this problem can be solved in time $O(\max(m,n,p))$. One important issue, the algorithm partition of the VLSI design is discussed and formal verification of the proposed VLSI architecture is given. The proposed architecture will be useful for remote sensing, satellite signal processing, and other related areas. It can also be useful for other 'packing' related tasks and for real-time digital picture processing.

LIST OF REFERENCES

1. K. S. Fu, Syntactic Pattern Recognition with Application, Prentice-Hall, Inc., 1982.

2. A. Rosenfeld and A. C. Kak, Digital Picture Processing, Vol. 2, Academic Press, New York.

3. R. J. Offen, VLSI Image Processing, Collins Professional and Technical Books, William Collins Sons & Co. Ltd., 1985.

4. T. Pavlidis, Algorithms for Graphics and Image Processing, Computer Science Press, 1982.

5. D. H. Ballard and C. M. Brown, Computer Vision, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982.

6. R. E. Bellman, Dynamic Programming, Princeton, NJ: Princeton Univ. Press, 1975.

7. K. Q. Brown, "Dynamic programming in computer science," CMU Tech. Rep., February 1979.

8. L. J. Guibas, H. T. Kung and C. D. Thompson, "Direct VLSI implementation of combinational algorithms," Caltech. Conf. on VLSI, January 1979.

9. K. H. Chu and K. S. Fu, "VLSI architectures for high speed recognition of general context-free languages and finite-state languages," Proc. 9th Annual Int'l. Symp. Comput. Arch., Austin, TX, April 1982.

10. Y. T. Chiang and K. S. Fu, "Parallel parsing algorithms and VLSI implementations for syntactic pattern recognition," IEEE trans. on Pattern Anal. Machine Intell., May 1984.

11. H. D. Cheng and K. S. Fu, "Algorithm partition and parallel recognition of general context-free languages using fixed-size VLSI architecture," Pattern Recognition, Vol. 19, No. 5, 1986.

12. H. H. Liu and K. S. Fu, "VLSI arrays for minimum-distance classifications," in VLSI for Pattern Recognition and Image Processing, edited by K. S. Fu, Springer-Verlag, Berlin, Heidelberg, 1984.

13. H. D. Cheng and K. S. Fu, "VLSI architectures for string matching and pattern matching," to appear in Pattern Recognition, Vol. 20, No. 1, 1987.

14. H. D. Cheng and K. S. Fu, "VLSI architecture for dynamic time-warp recognition of hand-written symbols," IEEE Trans. Acoust., Speech, Signal Processing, Vol. 34, No. 3, June 1986.

15. M. J. Clarke and C. R. Dyer, "Curve Detection in VLSI," VLSI for Pattern Recognition and Image Processing, edited by K. S. Fu, Springer-Verlag, Berlin, Heidelberg, 1984.

16. W-M Chow and L. T. Kou. "Matching Two Digital Pictures," Proc. International Computer Symposium, December 1978.

omit

ROBOT SIMULATION AND CONTROL

S20-63
11P.
188140

Inverse Kinematic-Based Robot Control

W.A. Wolovich and K.F. Flueckiger
Brown University
Providence, RI 02912

B1720314

1. Introduction

A fundamental problem which must be resolved in virtually all non-trivial robotic operations is the well-known *inverse kinematic question*. More specifically, most of the tasks which robots are called upon to perform are specified in *Cartesian (x,y,z) space*, such as simple tracking along one or more straight line paths or following a specified surface with compliant force sensors and/or visual feedback. In all cases, control is actually implemented through coordinated motion of the various links which comprise the manipulator; i.e. in *link space*. As a consequence, the control computer of every "sophisticated" anthropomorphic robot must contain provisions for solving the inverse kinematic problem which, in the case of "simple", non-redundant position control, involves the determination of the first three link angles, θ_1, θ_2, and θ_3, which produce a desired wrist origin position p_{xw}, p_{yw}, and p_{zw} at the end of link 3 relative to some fixed base frame, as further explained in [1].

as below

It is well known that the forward kinematic equations (the functional dependence of p_{xw}, p_{yw}, and p_{zw} on the θ_i's) usually can be obtained in a relatively straightforward manner using (say) the Denavit-Hartenberg transformation matrices in order to obtain functions G_x, G_y, and G_z in the relation:

theta sub 1
sub 2
sub 3

G sub 1
G sub 2

$$\begin{bmatrix} p_{xw} \\ p_{yw} \\ p_{zw} \end{bmatrix} \cancel{\triangleq X =} \begin{bmatrix} G_x(\theta_1,\theta_2,\theta_3) \\ G_y(\theta_1,\theta_2,\theta_3) \\ G_z(\theta_1,\theta_2,\theta_3) \end{bmatrix} \triangleq G(\theta) \tag{1}$$

P sub xw, P sub yw, and P sub zw

~~However, the converse determination~~ of a particular

$$\theta \triangleq \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} \tag{2}$$

which "solves" (1) for a given or desired X is not nearly as straightforward, although analytical inverse kinematic solutions do exist for virtually all current industrial manipulators. It might be noted, however, that these analytical inverse kinematic solutions are usually non-unique and sequential, and further require the evaluation of some rather complex Atan2 functions---see Summary Sheet 3.4.57 of [1], for example, which presents one such solution for the PUMA 560 industrial manipulator.

We should also note that this problem becomes significantly more complex when the orientation question is addressed simultaneously; i.e. when a desired end effector or tool orientation is specified in addition to its position. Certainly, any technique which can "simplify" solutions to the inverse kinematic question in robotics can have a significant impact not only on the computational requirements involved with robot control, but also on the diversity of tasks the manipulator can perform. *The primary purpose of this paper will be to thoroughly evaluate, extend, and demonstrate a new computational technique for solving the complete configuration (position and orientation) inverse kinematic problem for a variety of multi-link manipulators.*

Researchers

~~In the next section, we~~ will outline ~~this~~ a new inverse kinematic solution and demonstrate its potential via some recent computer simulations. We ~~will~~ also compare it to current inverse kinematic methods and outline some ~~of the~~ remaining problems which will be addressed in order to render it fully operational. ~~In Section 3 we will discuss a~~ number of practical consequences of this technique beyond its obvious use in solving the inverse kinematic question. ~~In particular, we will show how a~~ number ~~of diverse but well known robotic problems may be handled by~~ suitable ~~modifications to and/or~~ extensions of this new inverse kinematic result.

Also discuss

They

2. A Complete Inverse Kinematic Solution

To motivate the more general, six degree of freedom solution to the inverse kinematic problem associated with the

PAGE 164 INTENTIONALLY BLANK

overall configuration of the end effector, we will first present a solution to the three degree of freedom inverse kinematic problem associated with only the position of (say) the wrist origin associated with the end of the third link of a six link manipulator. The particular solution given here follows directly from that given in [2] with J^T replaced by J^{-1}, as suggested in [2] and later implemented in [3], where J denotes the well known *Jacobian matrix* of the manipulator. More specifically, the time differentiation of (1) directly implies that

$$\dot{X} = \begin{bmatrix} \frac{\partial G_x}{\partial \theta_1} & \frac{\partial G_x}{\partial \theta_2} & \frac{\partial G_x}{\partial \theta_3} \\ \frac{\partial G_y}{\partial \theta_1} & \frac{\partial G_y}{\partial \theta_2} & \frac{\partial G_y}{\partial \theta_3} \\ \frac{\partial G_z}{\partial \theta_1} & \frac{\partial G_z}{\partial \theta_2} & \frac{\partial G_z}{\partial \theta_3} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} = J\theta , \tag{3}$$

with the Jacobian J being a matrix of partial derivatives, as specified via equation (3).

In light of the preceding, now consider the closed loop dynamical system depicted in Figure 1, which is "driven" by some desired wrist origin position in Cartesian space, namely

$$X_d = \begin{bmatrix} p_{xwd} \\ p_{ywd} \\ p_{zwd} \end{bmatrix} \tag{4}$$

It can be noted that in Figure 1, K might be a (3×3) arbitrary, diagonal, time-invariant gain matrix, $\dot{\theta}_s$ would be a time varying 3-vector system output which represents the derivative of the desired link angle displacement which, when integrated, yields the 3-vector output representative of the link angle displacement, θ_s , and G(·) represents the forward kinematic operator defined by equation (1).

We might next define the equations which describe the dynamical behavior of the Figure 1 system, namely

$$\dot{\theta}_s = J^{-1}(\theta_s)K(X_d - X_s) , \tag{5}$$

and

$$X_s = G(\theta_s) . \tag{6}$$

Clearly, the premultiplication of (5) by $J(\theta_s)$ and the subsequent substitution of $\dot{X}_s$ for $J(\theta_s)\dot{\theta}_s$, in light of (3), then implies that

$$\dot{X}_s = K(X_d - X_s) , \tag{7}$$

or that X_s has a dynamical system representation as depicted in Figure 2. The reader will immediately recognize the system of Figure 2 as a parallel combination of three relatively simple, decoupled, first order, linear, time invariant systems with arbitrarily adjustable (via the elements of K) stability properties. In particular, if X_d represents a step input of magnitude X_d (actually a 3-vector step input), applied at time t_0, then it is easy to show that for zero initial conditions on X_s,

$$X_s(t) = \left[I - e^{-K(t-t_0)}\right]X_d , \tag{8}$$

or that for K positive definite, $X_s(t)$ will track the desired Cartesian position $X_d(t) = X_d$ with an (arbitrarily fast) exponentially decaying error! In light of (6), it therefore follows that $\theta_s(t)$ can be made to approach the desired θ_d which corresponds to $X_d = G(\theta_d)$ arbitrarily fast as well.

The reader might next note that in order to make this inverse kinematic procedure applicable to more general forms of robotic motion, it has to be "extended" to include inverse orientation information as well; i.e. solutions for θ_4, θ_5, and θ_6 of a general six link manipulator. However, the extension of the Figure 1 system to include orientation as well as position is a non-trivial task, since (i) there is no 3-vector representation for orientation and (ii) even if there were, the Figure 1 system would then require an analytical expression for the inverse of a corresponding (6×6) Jacobian, a formidable computational task. In light of these observations, we will now present, for the first time, a complete dynamical system solution to the inverse kinematic problem for both position and orientation.

To begin, we first note that the orientation of (say) the tool frame relative to the fixed base frame can be, and often is, specified by an appropriate (3x3) orientation coordinate transformation matrix, often called a *rotation matrix*, of the form (using the notation in [1]):

$$R_0^t = \begin{bmatrix} a, n, s \end{bmatrix} = \begin{bmatrix} a_x & n_x & s_x \\ a_y & n_y & s_y \\ a_z & n_z & s_z \end{bmatrix}, \tag{9}$$

where the orthogonal unit vectors a, n, and s represent the *approach*, *normal*, and *sliding* vectors associated with the orientation of the tool frame relative to some fixed base frame. Furthermore, since s can be obtained via the vector cross-product relationship:

$$a \times n = s\,, \tag{10}$$

as described in [1], knowledge of a and n alone will uniquely specify orientation of the end effector.

We next note that if

$$\omega = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{11}$$

represents the angular velocity of the tool frame, then it is not difficult to show, in light of Figure 3, that ω can be represented by the sum of its "translational component" relative to the motion of a, namely the cross product $a \times \dot{a}$, where $\dot{a} = \frac{da}{dt}$, and its "rotational component" relative to the motion of a, namely the scalar velocity dot product $\dot{n} \cdot s$ multiplied by a; i.e.

$$\omega = a \times \dot{a} + (\dot{n} \cdot s)a\,. \tag{12}$$

Furthermore, it now follows by expanding (12) in light of (9) that ω is also given by the following matrix-vector product:

$$\omega = \begin{bmatrix} s_x a_x & s_y a_x & s_z a_x & 0 & -a_z & a_y \\ s_x a_y & s_y a_y & s_z a_y & a_z & 0 & -a_x \\ s_x a_z & s_y a_z & s_z a_z & -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} \dot{n}_x \\ \dot{n}_y \\ \dot{n}_z \\ \dot{a}_x \\ \dot{a}_y \\ \dot{a}_z \end{bmatrix} \triangleq F(\theta) \begin{bmatrix} \dot{n} \\ \dot{a} \end{bmatrix} \tag{13}$$

The results which now follow build on the material presented in Section 4.3 of [1] which pertains to so-called *spherical wrist manipulators*. In such cases, (4.3.2) of [1] establishes the fact that

$$\dot{\bar{X}} \triangleq \begin{bmatrix} \dot{p}_{xw} \\ \dot{p}_{yw} \\ \dot{p}_{zw} \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} \dot{p}_w \\ \omega \end{bmatrix} = J\dot{\theta} = \begin{bmatrix} J_P & 0 \\ J_{\bar{R}} & J_R \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \\ \dot{\theta}_5 \\ \dot{\theta}_6 \end{bmatrix}, \tag{14}$$

or that the (6×6) Jacobian matrix associated with spherical wrist manipulators can be "triangularized", with J_P a (3×3) "positional" Jacobian associated with the velocity of the wrist origin relative to motion of the first three links, and J_R a (3×3) "orientation" Jacobian associated with the angular velocity of the end effector frame relative to the motion of the final three links.

If we now "invert" (14), it follows that

$$\dot{\theta} = J^{-1}\begin{bmatrix}\dot{p}_w \\ \omega\end{bmatrix} = \begin{bmatrix} J_P^{-1} & 0 \\ -J_R^{-1}J_{\bar{R}}J_P^{-1} & J_R^{-1}\end{bmatrix}\begin{bmatrix}\dot{p}_w \\ \omega\end{bmatrix}, \tag{15}$$

or, in light of (13), that

$$\dot{\theta} = \begin{bmatrix} J_P^{-1} & 0 \\ -J_R^{-1}J_{\bar{R}}J_P^{-1} & J_R^{-1}\end{bmatrix}\begin{bmatrix} I_3 & 0 \\ 0 & F(\theta)\end{bmatrix}\begin{bmatrix}\dot{p}_w \\ \dot{n} \\ \dot{a}\end{bmatrix} \triangleq J_I\begin{bmatrix}\dot{p}_w \\ \dot{n} \\ \dot{a}\end{bmatrix}, \tag{16}$$

with the (6×9) "inverse" Jacobian, J_I, given by the product of the (6×6) triangular inverse Jacobian defined by (15) and the (6×9) "block diagonal" matrix consisting of an upper left (3×3) identity matrix I_3, and a lower right (3×6) $F(\theta)$, as defined via (13).

We next note that the 9-vector "configuration"

$$\underline{X} \triangleq \begin{bmatrix}p_w \\ n \\ a\end{bmatrix} = \underline{G}(\theta) \tag{17}$$

for a known $\underline{G}(\cdot)$, with corresponding

$$\dot{\underline{X}} = \begin{bmatrix}\dot{p}_w \\ \dot{n} \\ \dot{a}\end{bmatrix}. \tag{18}$$

As defined, $\underline{X}$ completely specifies both the (wrist origin) position[†] and the (end effector) orientation of any given manipulator.

Now consider the dynamical system depicted in Figure 4, which we claim "solves" the inverse kinematic problem associated with the complete configuration of six link, spherical wrist manipulators. In particular, the dynamical equations associated with Figure 4 are

$$\dot{\theta}_s = J_I(\theta_s)K\left[\dot{\underline{X}}_d - \underline{X}_s\right], \tag{19}$$

with K a diagonal (9×9) gain matrix, and

$$\underline{X}_s = \underline{G}(\theta_s). \tag{20}$$

Since $\dot{\theta}_s$ is also equal to $J_I(\theta_s)\dot{\underline{X}}_s$, in light of (16) and (18), with $\dot{\underline{X}}_s$ arbitrary, (19) implies that

$$\dot{\underline{X}}_s = K\left[\underline{X}_d - \underline{X}_s\right], \tag{21}$$

or that the 9-vector $\underline{X}_s$ is analogous to the 3-vector X_s of (7). This in turn implies that $\underline{X}_s$ will track the desired Cartesian configuration $\underline{X}_d$ with an (arbitrarily fast) exponentially decaying error! As we noted earlier, it therefore follows that $\theta_s(t)$ can be made to approach the desired θ_d which corresponds to $\underline{X}_d = \underline{G}(\theta_d)$ arbitrarily fast as well. In other words, *the Figure 4 dynamical system solves for the first time the complete inverse kinematic problem for virtually any six link, spherical wrist manipulator.*

Figures 5 and 6 depict actual simulated runs of the Figure 4 system for the PUMA 560 industrial manipulator, as mathematically described in [1], when the (end effector) position vector (p_x, p_y, p_z) goes from (3.5, 2.5, 2.9) at $t_0 = 0$ to (1.5, 2.0, 4.4) at $t_f = 5$ along a LSPB (Linear Segment with Parabolic Blend) trajectory[‡] while the orientation of the end effector frame undergoes a simultaneous smooth transition for $(n_x, n_y, n_z, a_x, a_y, a_z)$ from (0, 0, -1, 0, 1, 0) to (-1, 0, 0, 0, 0, -1) over the same 5 second time interval. Only four of the nine elements of $\underline{X}$ are explicitly depicted, and in both cases,

[†] The term *link space* rather than *joint space* will be used here for reasons which are delineated in Section 1.4 of [1].

† Of course, for spherical wrist manipulators, knowledge of the wrist origin position and a, the approach vector, directly implies knowledge of the

the initial conditions on θ_s were appropriately set to insure that $\underline{X}_s(0) = \underline{X}_d(0)$. It might be noted that in the Figure 5 runs all nine of the nonzero, diagonal elements of K were set equal to 10, while these same nine elements were increased to 100 for the Figure 6 runs. The reader will note that a rather small error exists between the desired and simulated dynamical system configuration parameters depicted in the K=10 case. Moreover, this small error is essentially eliminated by increasing the (elements of the) diagonal gain matrix K to 100, as depicted in Figure 6; i.e. the desired and simulated configuration parameters are so close that they are virtually undistinguishable in this latter case! This, in turn, implies corresponding dynamical system link output displacement values which are "very close" to those which would mathematically solve the inverse kinematic question for the given, desired $\underline{X}_d = \begin{bmatrix} p_{wd} \\ n_d \\ a_d \end{bmatrix}$, especially in the K=100 case. In summary, therefore, *Figures 5 and 6 clearly illustrate the employment of the Figure 4 dynamical system as a viable alternative technique for solving the inverse kinematic question for a large class of multi-link manipulators.*

A number of observations are now in order relative to this dynamical system inverse kinematic solution. First of all, we note that $\dot{\theta}_s$ is also obtained as an output of our dynamical system solution *without explicit knowledge or use of* $\dot{\underline{X}}$! This could prove most useful in the implementation of a variety of control schemes which require desired link velocities as well as positions; e.g. in relatively simple PID controllers, where D denotes the (time) derivative of the link positional drive signal.

We next note that because of the spherical wrist assumption, we actually can determine an analytical expression for the (6×9) "inverse" Jacobian, $J_I(\theta)$, as defined by equation (16). For example, such an analytical expression is essentially given in Example 4.3.23 of [1] for the PUMA 560 industrial manipulator. Certain earlier reports and texts have implied that analytical expressions for J^{-1} in the six link case are virtually impossible to obtain. In [1] we show that this is not necessarily the case for spherical wrist manipulators, and here we exploit this observation to extend a three-dimensional inverse kinematic positional result to the more general and important, six-dimensional configuration case.

We further note that the particular inverse kinematic (position and velocity) solutions we obtain via the dynamical system of Figure 4 will be *unique*, and will depend on the initial conditions associated with the system. Different initial conditions can be used to produce all of the solution sets associated with a given manipulator, if desired, or only the particular one "best suited" to a specific task, such as (say) an arm right, elbow above trajectory for the PUMA 560 (see Figure 3.4.56 of [1]).

We finally observe, again in light of Figures 5 and 6, that there is no need to sequentially solve a set of rather complex and time-consuming Atan2 functions associated with a given robot to obtain the inverse kinematic link displacements associated with a desired Cartesian configuration. Although the computational savings associated with the direct employment of the Figure 4 dynamical system, rather than the explicit solution of a sequential set of Atan2 functions, has yet to be completely determined, there is reason to believe that such savings can be rather significant.

3. Practical Consequences to be Investigated

There are numerous practical consequences associated with the new computational inverse kinematic procedure which has just been outlined, and the primary purpose of this section will be to delineate some of them. To begin, we might again note the obvious, namely that the procedure can be directly utilized *to produce desired link positional and velocity drive signals to the link motors* which then might be controlled by any "standard procedure", such as a unity feedback PID compensator, without the explicit evaluation of any analytical Atan2 functions. Of course, in such cases and in the others which we will outline in this section, it is important to realize that a flexible control computer must be employed in order to physically realize (say) the Figure 4 feedback system. In light of this observation, it is of interest to note that a truly significant amount of robotics development effort within LEMS at Brown University over the past year has focused on the development of one such flexible control computer for robotic applications, namely SIERA (System for Implementing and Evaluating Robotic Algorithms).

SIERA is a unique multiprocessor system composed of two subsystems--a tightly-coupled real-time servo system and a loosely coupled multiprocessor network (the "Armstrong system"), as depicted in Figure 7. A shared memory

end effector position as well, as is shown in [1].

‡ Both references [1] and [6] describe such LSPB trajectories.

interface allows communication between these two subsystems. The architecture is flexible enough to accommodate a variety of robots and sensors, since all robot dependent hardware is restricted to the robot interface board. Thus, we have been able to control the Unimation Puma 560 and the IBM 7565 robots that are currently in our laboratory. A detailed description of the SIERA hardware can be found in [4].

The SIERA operating system provides a flexible development system for research in robotic algorithms, without making the system too complex to be used for instructional purposes. This is accomplished by defining three different programming levels: i) the user level, which is analogous to a commercial system such as Unimation's VAL robot command language, ii) the researcher level, which fulfills the main objective of SIERA by allowing any type of robotic algorithm to be added to the system, and iii) the expert level, which is used to add a new robot or to enhance the operating system. It should be noted that the operating system is also generally applicable since all (low-level) robot tasks are handled by interface routines written by an expert level programmer. Further details of the operating system and the programming levels can be found in [5].

Another potential use for our inverse kinematic procedure which has yet to be fully exploited is in *the automatic avoidance of degenerate configurations*, such as those associated with Jacobian singularities. To be more specific, it is well known that certain desired Cartesian trajectories may imply corresponding link trajectories for which $|J(\theta)|$, the determinant of the Jacobian, approaches zero. In such cases, excessive link velocities are required to produce seemingly well-behaved Cartesian motion. We feel that one way of automatically avoiding such degenerate configurations could be to physically restrict the magnitude that $|J(\theta_s)|$ can decrease to in either the Figure 1 or the Figure 4 system. Although such a procedure will not yield the desired Cartesian trajectories, hopefully it will yield "acceptable" Cartesian trajectories which are "close to" the specified ones. Some preliminary computer simulations bounding $|J(\theta_s)|$ have produced rather encouraging results, and one of the primary objectives of our continuing research will be to thoroughly investigate this and other automatic degenerate configuration avoidance techniques.

Another potential use of our inverse kinematic procedure is that associated with *redundant manipulators*; i.e. manipulators which have more degrees of freedom than are necessary to achieve (say) desired end effector orientations. To be more specific, it is well known that the inverse kinematic problem associated with redundant manipulators can have an infinite number of solutions, and the problem then becomes one of appropriately selecting the "best" solution from this infinite set. It might be noted that one way of obtaining a variety of different link solutions, (say) in light of Figure 1, is to employ "different right inverse" Jacobians instead of the square $J^{-1}(\theta_s)$ depicted. Our investigations are continuing to determine how a "best right inverse" Jacobian might be selected and utilized in our computational inverse kinematic procedure in order to automatically yield a correspondingly "best" inverse kinematic solution for redundant manipulators.

Another potentially important application of our computational inverse kinematic procedure concerns *its employment in more sophisticated control strategies* where knowledge of $\ddot{\theta}_s(t)$, as well as $\dot{\theta}_s(t)$ and $\theta_s(t)$, would be used. One such example is that associated with the inverse dynamic, feedforward compensation procedure outlined in Section 8.5 of [1]. We have already conducted some preliminary simulations of an "extended" version of the Figure 1 and Figure 4 dynamical systems ("extended" by the addition of another parallel bank of integrators as well as appropriate feedback gain matrices) in order to produce $\ddot{\theta}_s$ as well as $\dot{\theta}_s$ and θ_s. One such "extended" system is depicted in Figure 8 in its simplest (positional) form. The mathematical equations associated with such a dynamical system can directly be shown to imply an analogous linear, time-invariant, second order differential equation relationship between input $X_d(t)$ and output $X_s(t)$, namely

$$\ddot{X}_s(t) + A\dot{X}_s(t) + KX_s(t) = X_d(t)\ , \tag{22}$$

which can then be used to establish convergence relations between $\theta_s(t)$ and its derivatives and the desired $\theta_d(t)$ and its derivatives. Results in this area are still under development. In particular, we are currently working on a more complete mathematically understanding of the Figure 8 system, including the implications regarding the $\ddot{\theta}_s(t)$, $\dot{\theta}_s(t)$, and $\theta_s(t)$ thus obtained, when compared to the desired values of $\theta_d(t)$ and its derivatives in both the simple (positional) case depicted and the full six degree of freedom configuration case. Here again, our initial simulations have been encouraging and we are actively continuing these investigations.

4. Summary

We have now outlined a new computational procedure for solving the inverse kinematic question for a large class of multi-link manipulators. Furthermore, we have mathematically established the "equivalence" between this computational procedure and the behavior of relatively simple first and second order, linear, time-invariant dynamical systems. We have indicated a number of potential practical consequences associated with the employment of this technique in robotic applications, namely:
(i) its use in directly obtaining unique values for the inverse kinematic positions, velocities, and accelerations,
(ii) its potential for automatically avoiding degenerate configurations,
(iii) its ability to produce the "best" inverse kinematic solutions for redundant manipulators, and
(iv) its employment in more sophisticated motion control strategies.

We have expended a considerable amount of time and effort within LEMS in constructing a general purpose, flexible robot control system (SIERA) which can be used to thoroughly implement, test, and evaluate all aspects of our robotics research program, and we have two industrial manipulators (a PUMA 560 anthropomorphic robot and an IBM RS/1 Cartesian robot) to employ in our studies. Our investigations are well underway, and we are very optimistic that significant new techniques for robot control and manipulation will result as a consequence of these investigations.

References

[1] Wolovich, W. A., *Robotics: Basic Analysis and Design*, Holt, Rinehart and Winston, 1986.

[2] Elliott, H. and Wolovich, W. A., *A Computational Technique for Inverse Kinematics*, Proceedings of the 1984 Conf. on Decision and Control, Las Vegas, NV Dec. 1984.

[3] Vaccaro, R. J. and Hill, S. D., "A Joint-Space Command Generator for Cartesian Control of Robotic Manipulators," Department of Electrical Engineering, University of Rhode Island, 1986, (to appear).

[4] Kazanzides, P., Wasti, H., Weinstein, R., and Wolovich, W. A., "SIERA: System for Implementing and Evaluating Robotic Algorithms," Brown University Technical Report LEMS-23, June, 1986. Also to be presented at the 1987 IEEE International Conference on Robotics and Automation in Raleigh, NC March 30 - April 3, 1987.

[5] Kazanzides, P., Wasti, H., and Wolovich, W. A., "SIERA: A Multiprocessor System for Robotics," Proceedings of the SPIE Symposium on Advances in Intelligent Robotic Systems, Cambridge, Mass., Oct. 1986.

[6] Craig, John J., *Introduction to Robotics: Mechanics & Control*, Addison-Wesley Publishing Company, 1986.

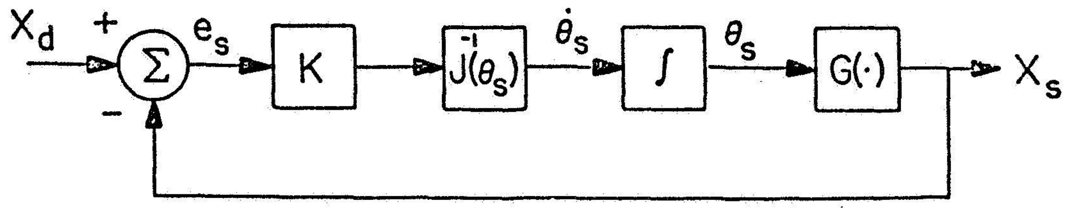


Figure 1
A Positional Inverse Kinematic Solution

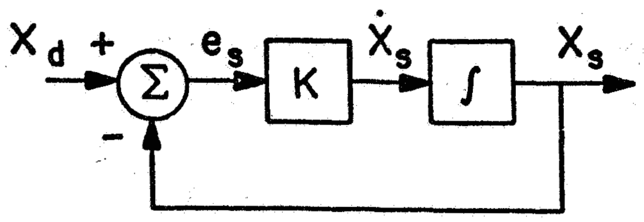


Figure 2
A Dynamical System Representation for X_s

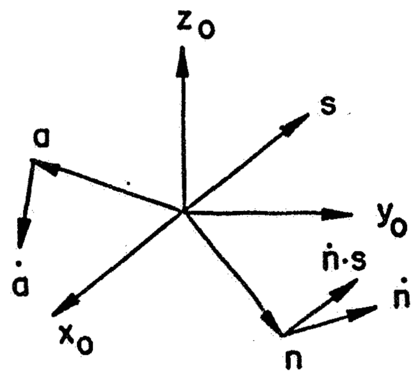


Figure 3
Rotation of the Tool Frame Relative to the Base Frame

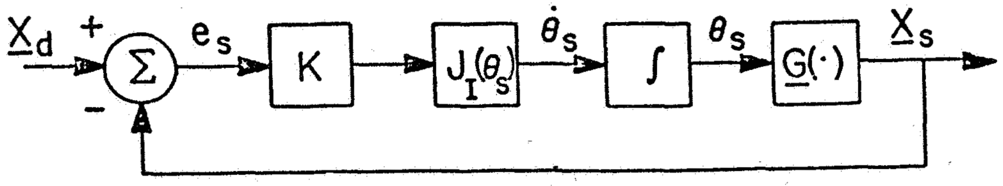


Figure 4
A Configuration Inverse Kinematic Solution

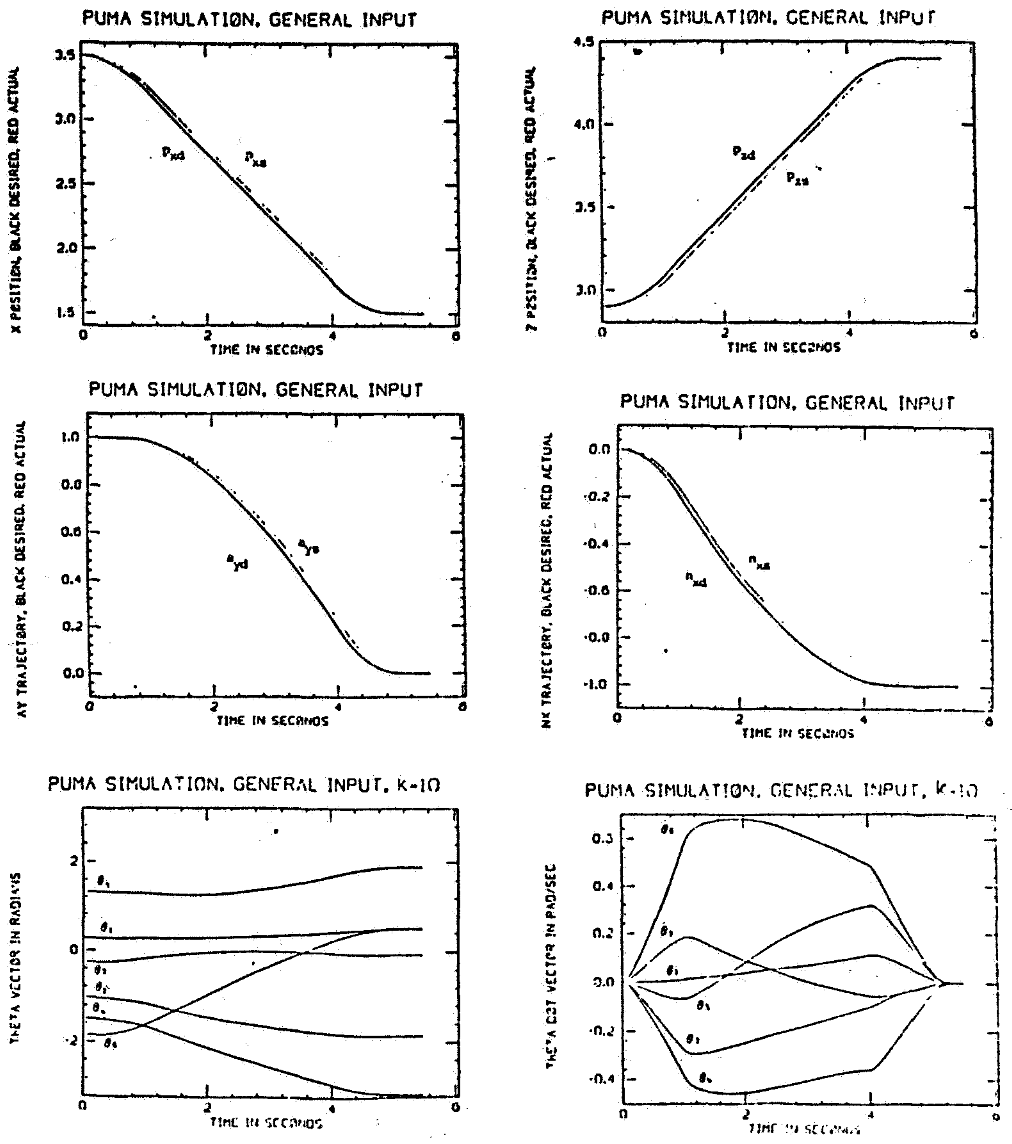


Figure 5
Puma Simulation with K=10

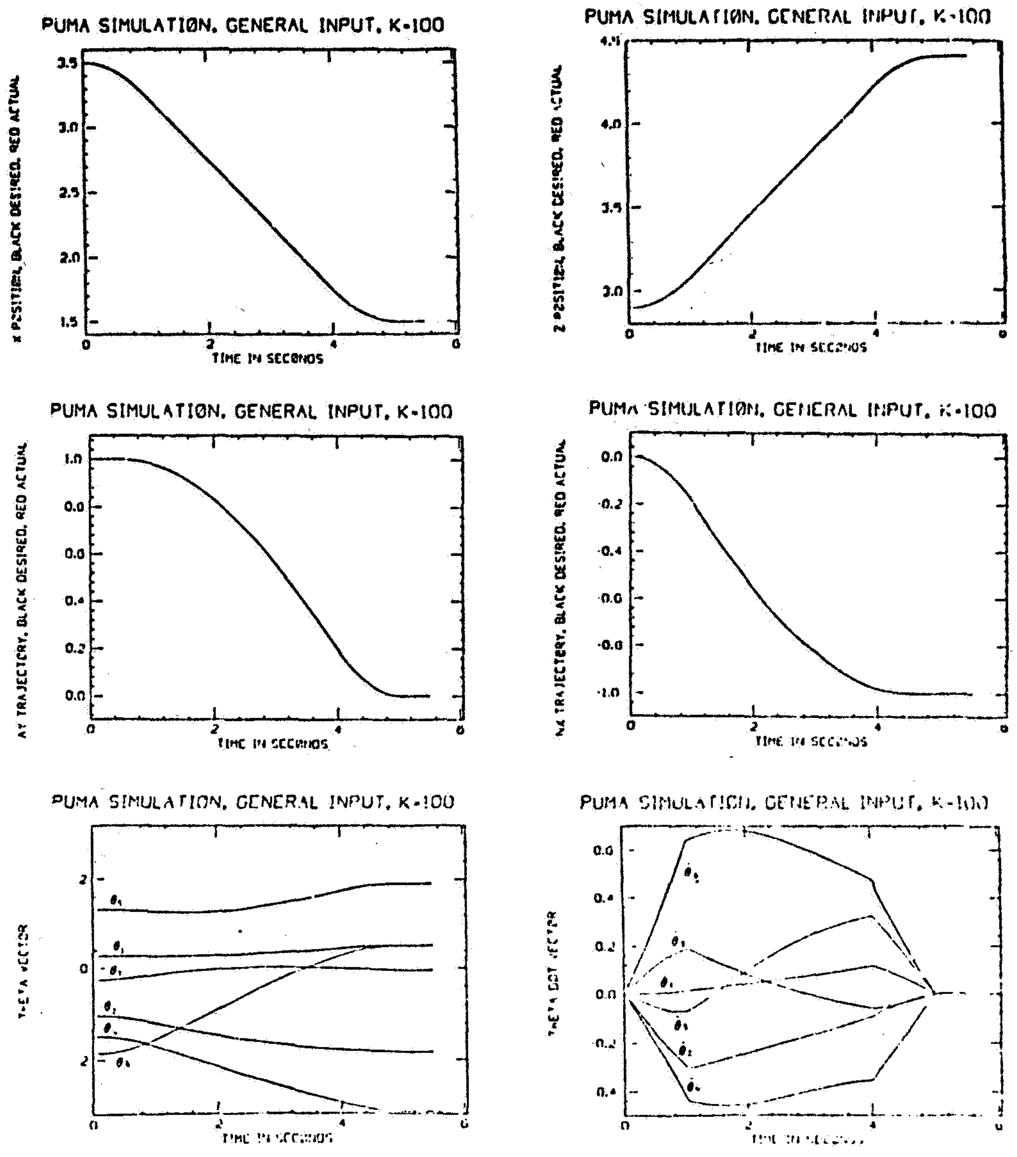


Figure 6
Puma Simulation with K=100

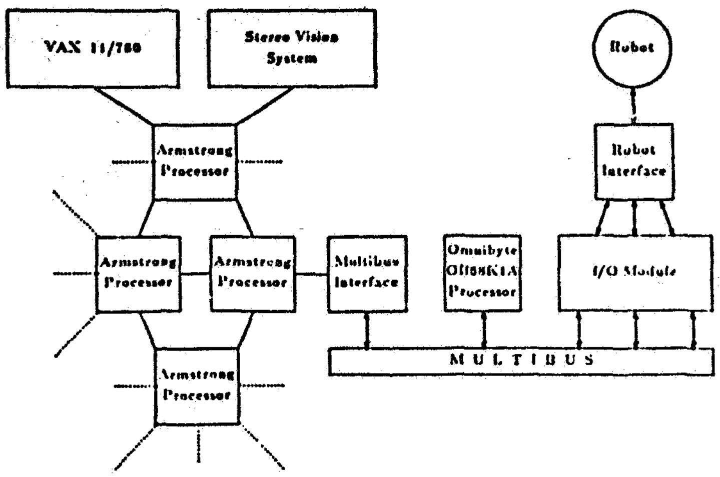


Figure 7. Overview of SIERA.

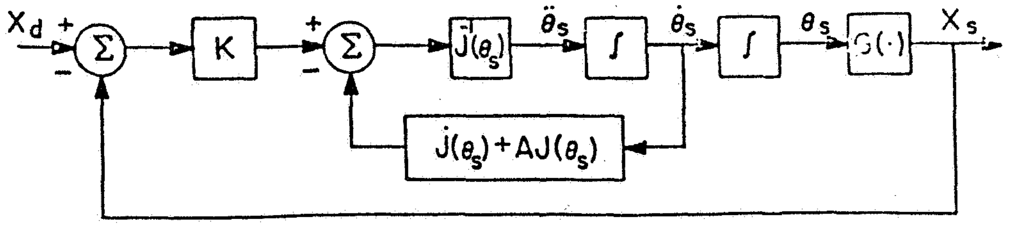


Figure 8
An "Extended" Positional Inverse Kinematic Solution

S21-37
8P.
188141

Spatially Random Models, Estimation Theory, and Robot Arm Dynamics

G. Rodriguez
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109

CB 553097

ABSTRACT

a/1855

Spatially random models provide an alternative to the more traditional deterministic models used to describe robot arm dynamics. These alternative models can be used to establish a relationship between the methodologies of estimation theory and robot dynamics. A new class of algorithms for many of the fundamental robotics problems of inverse and forward dynamics, inverse kinematics, etc. can be developed that use computations typical in estimation theory. The algorithms make extensive use of the difference equations of Kalman filtering and Bryson-Frazier smoothing to conduct spatial recursions. The spatially random models are very easy to describe and are based on the assumption that all of the inertial (D'Alembert) forces in the system are represented by a spatially distributed white-noise model. The models can also be used to generate numerically the composite multibody system inertia matrix. This is done without resorting to the more common methods of deterministic modeling involving Lagrangian dynamics, Newton-Euler equations, etc. These methods make substantial use of human knowledge in derivation and manipulation of equations of motion for complex mechanical systems. In contrast, with the spatially random models, more primitive (i.e., simpler and less dependent on mathematical derivations) locally specified computations result in the emergence of a global collective system behavior equivalent to that obtained with the deterministic models.

1. INTRODUCTION

Recently, an equivalence has been discovered between estimation theory and recursive robot arm dynamics [1], as summarized in the following table.

TABLE 1

Equivalence Between Optimal Estimation
and
Recursive Robot Arm Dynamics

ESTIMATION		ROBOT DYNAMICS
States	$x(k)$	Spatial Forces
Co-States	$\lambda(k)$	Spatial Accelerations
Measurements	$\tau(k)$	Joint Moments
Transition Matrix	$\phi(k,k-1)$	Spatial Jacobian
Process Error Covariance	$M(k)$	Spatial Inertia Matrix
Known Input	$b(k)$	Bias Spatial Force
State-to-Output Map	$H(k)$	State-to-Joint-Axis Map

A spatial force $x(k)$ is a 6-dimensional vector consisting of three pure moment components and three force components. The argument k refers to a representative body k in a multibody system. Similarly, $\lambda(k)$ is a 6-dimensional vector of three angular acceleration components and three linear acceleration components. The joint moments $\tau(k)$ are due to external sources acting at the joints. The spatial transition matrix serves

PAGE 176 INTENTIONALLY BLANK

to propagate spatial forces within a body [1] in an inward direction from joint $k-1$ to joint k. Its transpose serves to propagate spatial accelerations in an opposite direction. The 6-by-6 matrix $M(k)$ represents the spatial inertia of body k about joint k. The state-to-output map $H(k)$ is a 1-by-6 vector that projects the spatial force into its component along the joint axis. The bias force $b(k)$ is due to nonlinear velocity and gravity dependent effects [1].

The spatial inertia matrix and the transition matrix associated with this system are defined as

$$M(k) = \begin{pmatrix} I(k) & m(k)\tilde{p}(k) \\ -m(k)\tilde{p}(k) & U \end{pmatrix}$$

$$\phi(k, k-1) = \begin{pmatrix} U & \tilde{L}(k) \\ 0 & U \end{pmatrix}$$

in which $I(k)$ is the body k inertia about joint k; $m(k)$ is the body k mass; $L(k)$ is the vector from joint k to joint $k-1$; and $p(k)$ is the vector from joint k to the body k mass center. The symbol U denotes the 3-by-3 identity.

A spatially random state space model for the multibody system is

$$x^-(k) = \phi(k, k-1)x^+(k-1) + \omega(k) \tag{1.1}$$

$$x^+(k) = x^-(k) \tag{1.2}$$

in which $x^-(k)$ is the value of the spatial force on the negative side of joint k, and $x^+(k-1)$ is the value of the spatial force on the positive side of joint $k-1$. The "+" superscript indicates that the corresponding force is evaluated at a point immediately adjacent to joint k and toward the base of the multibody system. Similarly, the "−" superscript indicates that the corresponding variable is evaluated on the negative side of joint k. Note that $x^+(k-1)$ and $x^-(k)$ refer to spatial forces that are acting on body k due to the adjacent bodies $k-1$ and $k+1$ respectively. Equation (1.2) expresses continuity of the spatial force in crossing a joint connecting two adjacent bodies.

The above is a linear model that reflects a balance of the forces that are acting on body k. The inertial forces are represented by a spatial white-noise process whose mean and covariance are

$$E[\omega(k)] = b(k) \quad \text{and} \quad E[\hat{\omega}(k)\hat{\omega}(k)^T] = M(k) \tag{1.3}$$

with $\hat{\omega}(k) = \omega(k) - b(k)$. The mean value of the inertial force $\omega(k)$ is set equal to the bias force $b(k)$. The covariance of the inertial force is set equal to the spatial inertia matrix. The output, or measurement, equation

$$\tau(k) = H(k)x^+(k) \tag{1.4}$$

completes a description of the stochastic model. In this model, the active joint moment $\tau(k)$ plays the role of the measurement in a linear state space system. Since the joint moments are known exactly, the corresponding measurement equation is free of measurement noise.

The above model can be cast in the more compact notation

$$X = \phi W \quad \text{and} \quad T = HX \tag{1.5}$$

where W, X, and T are the composite vectors $W = [\omega(1), \ldots, \omega(N)]$, $X = [x(1), \ldots, x(N)]$ and $T = [\tau(1), \ldots, \tau(N)]$. Here, N represents the total number of bodies in the multibody system. The composite process error vector W has a mean and covariance given by

$$E(W) = b \quad \text{and} \quad E[\widehat{W}\widehat{W}^*] = Q \tag{1.6}$$

where $b = [b(1), \ldots, b(N)]$. The $6N$-by-$6N$ block diagonal matrix Q is defined as $Q = diag[M(1), \ldots, M(N)]$. Its typical 6-by-6 diagonal block $M(k)$ is the spatial inertia of body k. The matrix ϕ is a causal (i.e., lower triangular) matrix defined as

$$\phi = \begin{pmatrix} U & 0 & \ldots & 0 \\ \phi(2,1) & U & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \phi(N,1) & \phi(N,2) & \ldots & U \end{pmatrix} \tag{1.7}$$

The closely related composite state-to-output map H in (1.5) is defined as $H = diag[H(1), \ldots, H(N)]$.

This model is now used to investigate a number of relationships between estimation theory and robot arm dynamics.

2. CONDITIONAL MEAN ESTIMATION

The estimation problem to be solved here is to estimate the process error vector W and the state X, given the measurements T. This corresponds to the dynamics problem of finding the inertial forces (due to accelerations) and the spatial forces, given the joint moments. The optimal estimates are obtained by means of the conditional expectations $E(W/T)$ and $E(X/T)$. It is relatively simple to compute these two conditional expectations, although care has to be exercised due to the non-zero mean of the inertial force W. By methods outlined in [2], it can be established that

$$E(X/T) = \phi b + G(T - H\phi b) \tag{2.1}$$

in which G is the "Kalman" gain

$$G = RH^*(HRH^*)^{-1} \quad \text{and} \quad R = \phi Q \phi^* \tag{2.2}$$

This is the estimate of the spatial forces given the applied joint moments. Note that the estimator equations have a predictor-corrector architecture. The prediction term is due to the bias force b in (2.1). This term "predicts" the cumulative spatial bias force on any given body due to the bias force acting on all of the preceding bodies. The covariance of the estimation error inherent in this "open-loop" predicted estimate is

$$E[(X - \phi b)(X - \phi b)^*] = \phi Q \phi^* = R \tag{2.3}$$

The prediction term is said to be open-loop because it is based only on the system model and does not depend on the measurement T. The effect of measurements is accounted for in the correction term involving the Kalman gain in (2.1). The Kalman gain determines the weight of the correction term, when this is added to the prediction term, to arrive at the final state estimate $E(X/T)$. The N-by-N matrix HRH^* that needs to be inverted to compute the Kalman gain turns out to be the composite multibody system inertia matrix.

To compute the covariance of the estimation error after correction has occurred, observe first that

$$X - E(X/T) = (I - GH)\phi W \tag{2.4}$$

is the estimation error. Its corresponding covariance is

$$P = (I - GH)R(I - GH)^* \tag{2.5}$$

Alternatively, this becomes

$$P = (I - GH)R = R(I - GH)^* = R - RH^*(HRH^*)^{-1}HR \tag{2.6}$$

Note that $HP = 0, PH^* = 0, HPH^* = 0$ which imply that the estimation error at the joints vanishes. This reflects the lack of measurement noise in the measurement Equation (1.4).

The conditional-mean estimate for the inertial forces is given by

$$E(W/T) = b + Q\phi^* H^*(HRH^*)^{-1}(T - H\phi b) \tag{2.7}$$

This estimate is made up of two elements. First is the element due to the bias force b. Second is the element due to the active moments T. To examine these two effects more closely, define the joint angle accelerations as

$$a = M^{-1}(T - H\phi b) \quad \text{where} \quad M = HRH^* \tag{2.8}$$

Observe that the matrix M, whose inversion is required to compute the joint angle accelerations, is the composite multibody system inertia matrix. In addition, observe [1] that the joint angle accelerations a and the spatial accelerations λ are related by

$$\lambda = \phi^* H^* a \tag{2.9}$$

Based on these definitions, the estimate for the inertial forces becomes

$$E(W/T) = b + Q\lambda \tag{2.10}$$

The covariance of the inertial force estimation error is obtained by arguments very similar to those used to arrive at (2.4). Observe first that $\widehat{W} = W - E(W/T) = [I - Q\phi^* H^*(HRH^*)^{-1}H\phi]W$ is the estimation error. Its covariance is

$$E[\widehat{W}\widehat{W}^*] = Q - Q\phi^* H^* M^{-1} H\phi Q \tag{2.11}$$

The foregoing are "batch" solutions to the estimation problem, in the sense that all of the measurements are processed simultaneously. This implies that the composite system inertia matrix must be inverted in a batch mode. An alternative is provided by the sequential solution outlined below.

3. SEQUENTIAL ESTIMATION

The sequential solution processes the measurements (the applied moments) one at a time. In doing this, it does not require numerical inversion of the N-by-N system inertia matrix. Instead, the inertia matrix is factored as

$$M = (I + K)D(I + K^*) \tag{3.1}$$

in which D is an N-by-N diagonal matrix, and K is a lower-triangular matrix. The matrices K and D in this factorization are generated using a suitably defined Kalman filter. This factorization of a covariance into a product of a causal factor, a diagonal matrix, and the anti-causal adjoint factor is strongly reminiscent of the celebrated [5] Gohberg-Krein factorization. Applications of this result to estimation problems have been investigated by Kailath [6]. Once this factorization of the system inertia matrix is achieved, the corresponding inverse can be computed easily. The central result is that

$$(I + K)^{-1} = I - L \tag{3.2}$$

where L is a lower- triangular causal matrix generated by the same Kalman filter that generates K. This implies that the inertia matrix inverse can be expressed as

$$M^{-1} = (I - L^*)D^{-1}(I - L) \tag{3.3}$$

The central aim of this section is to outline how to obtain this result. Only the major results are presented. The detailed arguments leading to the results will be presented elsewhere by the author.

Result 3.1. *The state covariance matrix $R = \phi Q \phi^*$ can be expressed as*

$$R = r + \Phi r + r\Phi^* \tag{3.4}$$

Here, Φ is the system model matrix obtained by subtracting the $6N$-by-$6N$ identity from the matrix in (1.7). The matrix r is a $6N$-by-$6N$ block diagonal matrix $r = diag[r(1), \ldots, r(N)]$ whose blocks $r(k)$ satisfy the recursive relationships

$$\begin{gathered} r^+(0) = 0 \\ r^-(k) = \phi(k, k-1)r^+(k-1)\phi^T(k, k-1) + M(k) \\ r^+(k) = r^-(k) \end{gathered} \tag{3.5}$$

Define now the block-diagonal matrix $P = diag[P(1), \ldots, P(N)]$ whose diagonal blocks $P(k)$ satisfy the discrete Riccati equation

$$\begin{gathered} P^-(k) = \phi(k, k-1)P^+(k-1)\phi^T(k, k-1) + M(k) \\ D(k) = H(k)P^-(k)H^T(k) \\ P^+(k) = P^-(k) - P^-(k)H^T(k)H(k)P^-(k)/D(k) \end{gathered} \tag{3.6}$$

with the "initial" condition $P^+(0) = 0$.

Result 3.2. *The matrices R and P above are related by*

$$R = P + \Phi P + P\Phi^* + \Phi PHD^{-1}HP\Phi^* \tag{3.7}$$

Result 3.3. *The system inertia matrix M factors as in (3.1), with the causal matrix K and the diagonal matrix D defined as*

$$K = H\Phi PH^*D^{-1} \quad \text{and} \quad D = HPH^* \tag{3.8}$$

Define now the transition matrix $\psi(k,m)$ by means of the Kalman filtering equations

$$\begin{aligned} \psi(k,k) &= I \\ \psi(k^-, k-1^+) &= \phi(k, k-1) \\ \psi(k^+, k^-) &= I - g(k)H(k) \end{aligned} \tag{3.9}$$

in which $g(k)$ is the Kalman gain

$$g(k) = P^-(k)H^T(k)D^{-1}(k) \tag{3.10}$$

Define also the related composite matrix

$$\Psi = \begin{pmatrix} 0 & 0 & \dots & 0 \\ \psi(2,1) & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \psi(N,1) & \psi(N,2) & \dots & 0 \end{pmatrix} \tag{3.11}$$

These two definitions can be used to establish the following identity.

Result 3.4. *The "open-loop" and "closed-loop" transition matrices Φ and Ψ are related by*

$$\Psi = (I - gH)\Phi \tag{3.12}$$

where $g = [g(1), \dots, g(N)]$ is the matrix of Kalman gains.

Result 3.5. *The lower triangular factor $I + K$ can be inverted as*

$$(I + K)^{-1} = I - L \tag{3.13}$$

in which L is the lower triangular matrix

$$L = H\Psi PH^*D^{-1} \tag{3.14}$$

This also implies that $K = L + KL$, $K = L + LK$, and $LK = KL$.

The above sequence of results is the necessary ingredient to establish the recursive factorization of the inverse of the composite system inertia matrix as in (3.3).

4. FILTERING AND SMOOTHING

Typically, the composite system inertia matrix is inverted to solve what is referred to as the forward dynamics problem. This problem consists of computing a set of joint angle accelerations given a corresponding set of applied joint moments. The joint angle accelerations a and the applied joint moments T are related by

$$a = (I - L^*)D^{-1}(I - L)T \tag{4.1}$$

where $a = [a(1), \dots, a(N)]$ is the vector of joint angle accelerations. This states that the joint moments must be processed by means of a two-stage computation. The first stage represents filtering and is characterized by the factor $(I - L)$.

The second stage represents smoothing and is characterized by the factor $(I - L^*)$.

Filtering. This stage produces an "innovations" process defined as

$$e^- = (I - L)T \tag{4.2}$$

It produces also the filtered state estimate

$$Z = \Psi P H^* D^{-1} T = \Psi g T \tag{4.3}$$

The components $z(k)$ of $Z = [z(1), \ldots, z(N)]$ satisfy the Kalman filter equations [1]

$$z^-(k) = \phi(k, k-1) z^+(k-1) + b(k) \tag{4.4}$$

$$z^+(k) = z^-(k) + g(k) e^-(k) \tag{4.5}$$

in which $e^-(k)$ are the elements of the innovations vector $e^- = [e^-(1), \ldots, e^-(N)]$. Multiplication of the innovations process by the inverse of the diagonal matrix D produces the residuals

$$e^+ = D^{-1} e^- \tag{4.6}$$

These residuals are processed in the smoothing stage that follows.

Smoothing. This corresponds to multiplication of the residuals by the anti-causal factor $(I - L^*)$ to obtain the joint angle accelerations, i.e.,

$$a = (I - L^*) e^+ \tag{4.7}$$

A spatial difference equation which is based on (4.7) can be obtained by re-introducing the co-state variables defined earlier. The co-state variables λ and the residuals e^+ are related by

$$\lambda = \Psi^* H^* e^+ \tag{4.8}$$

Use of this in (4.7) implies that

$$a = e^+ - g^* \lambda \tag{4.9}$$

This last relationship expresses the joint angle accelerations in terms of the residuals and the co-state variables. Furthermore, (4.8) can be used to infer that the co-state variables satisfy the difference equation

$$\lambda^+(k-1) = \phi^T(k, k-1) \lambda^-(k) \tag{4.10}$$

$$\lambda^-(k) = \lambda^+(k) + H^T(k) e^+(k) \tag{4.11}$$

with the terminal condition $\lambda^+(N) = 0$. These equations are referred to as the Bryson-Frazier smoother equations [4]. Their application to problems in robot dynamics is discussed in more detail in [1].

5. COVARIANCE ANALYSIS

The aim here is develop formulas to compute the covariance of several relevant quantities (state, state estimation error, innovations, etc.) discussed in previous sections. The stochastic model (1.5) is assumed as a starting point. As in earlier discussions, the results are stated without proof.

Result 5.1. *The composite system inertia matrix M is the covariance of the measurement process, i.e.,*

$$M = E(TT^*) = HRH^* \tag{5.1}$$

This result has an interesting interpretation. It states that the collective system behavior, as represented by the system inertia, emerges from the covariance of the output T of the spatially random model (1.5). It therefore provides a means to compute the inertia matrix numerically by direct simulation of the stochastic model. From such a simulation, the inertia matrix would emerge (without conducting the more traditional manual derivation of the equations of motion).

Result 5.2. *The spatial inertia matrix P produced by the Riccati equation is equal to the covariance of the state estimation error, i.e.,*

$$E[(X-Z)(X-Z)^*] = P + \Psi P + P\Psi^* \tag{5.2}$$

The corresponding mean-square estimation error is

$$E[(X-Z)^*(X-Z)] = Tr[P] \tag{5.3}$$

Result 5.3. *The innovations process has a covariance given by*

$$E[(e^-)(e^-)^*] = D \tag{5.4}$$

Result 5.4. *The covariance of the co-states is*

$$E(\lambda\lambda^*) = \Psi^* H^* D^{-1} H \Psi = \Lambda + \Lambda\Psi + \Psi^*\Lambda \tag{5.5}$$

in which $\Lambda = diag[\Lambda(1), \ldots, \Lambda(N)]$. The diagonal blocks $\Lambda(k)$ satisfy

$$\Lambda^-(k) = [I - g(k)H(k)]^T \Lambda^+(k)[I - g(k)H(k)] + H^T(k)H(k)/D(k) \tag{5.6}$$

$$\Lambda^+(k-1) = \phi^T(k, k-1)\Lambda^-(k)\phi(k, k-1) \tag{5.7}$$

with the terminal condition $\Lambda^+(N) = 0$.

6. CLOSED-FORM INERTIA MATRIX INVERSE

The foregoing results can be used to obtain in closed form the inverse of the composite multibody system inertia. This is done in terms of the covariance matrices P and Λ of the previous section.

Result 6.1. *The inverse of the system inertia matrix can be expressed as*

$$M^{-1} = D^{-1} + g^*\Lambda g + g^*\Psi^*(\Lambda g - H^* D^{-1}) + (g^*\Lambda - D^{-1}H)\Psi g \tag{6.1}$$

Alternatively, it can be expressed as $M^{-1} = D^{-1}HSH^*D^{-1}$ where

$$S = P + P\Lambda P + P\Psi^*(\Lambda P - I) + (P\Lambda - I)\Psi P \tag{6.2}$$

This result is quite similar to that obtained in [1] by more detailed methods. The result has an interesting potential application in robot dynamics analysis and in control design. The equations of motion for the multibody system representing a robot arm are typically written, neglecting bias forces and accelerations due to nonlinear velocity and gravity dependent effects, in the form

$$Ma = T \tag{6.3}$$

where a is the set of joint angle accelerations, and T is the set of applied joint moments. The primary reason for the widespread use of such an equation is that many of the known methods for deriving equations of motion result in a matrix equation of this form. The equation consistently involves the presence of a composite system inertia matrix. There is, however, nothing intrinsic in the multibody dynamics problem that would make the presence of an inertia matrix in the equations of motion completely inevitable. In fact, Result 6.1 shows how to compute the inverse of the inertia matrix directly, without having to evaluate the inertia matrix first and then having to invert it. It is therefore possible, by using this result, to arrive directly, without numerical inversion of the inertia matrix, at a set of motion equations of the form

$$a = M^{-1}T \tag{6.4}$$

This is potentially a very useful result, since the system in (6.4) is much easier to work with, in simulation and control design, for instance, than the equivalent system in (6.3).

7. CONCLUDING REMARKS

The use of spatially distributed random models has been explored in analyzing robot arm dynamics. Based on such models a previously undiscovered relationship has been established between estimation theory and recursive robot arm dynamics. Many of the fundamental problems in robot dynamics can be approached using the techniques of estimation theory. The interaction between these two areas has not been recognized before and leads to many useful insights, such as the equivalence of covariance and spatial inertia. The numerical properties of the new algorithms emerging from the estimation approach to robot dynamics are under investigation.

ACKNOWLEDGMENT

This paper reports on one phase of research performed at the Jet Propulsion Laboratory, California Institute of Technology under the sponsorship of the National Aeronautics and Space Administration.

REFERENCES

[1] G. Rodriguez, "Kalman Filtering, Smoothing and Recursive Robot Arm Forward and Inverse Dynamics," *JPL Publication* 86-48, Jet Propulsion Laboratory, Pasadena, CA. (Dec. 1986).

[2] A. V. Balakrishnan, Applied Functional Analysis, *Springer-Verlag* (1977).

[3] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *ASME Trans., J. Basic Engr. Vol. D* (March 1960), 35-45.

[4] A. E. Bryson, Jr. and Y. C. Ho, Applied Optimal Control, *Blaisdell Publishing Co.* (1969).

[5] I. C. Gohberg and M. G. Krein, "Theory and Applications of Volterra Operators in Hilbert Space," *American Mathematical Society Translations* (1970).

[6] T. Kailath, "An Innovations Approach to Least-Squares Estimation," *IEEE Transactions on Information Theory Vol.* 11 (1968), 646-655.

S22-37
118.
188142

Task Oriented Nonlinear Control Laws for Telerobotic Assembly Operations

R.A. Walker,[†] L.S. Ward,[*] and C.F. Elia[*]
Integrated Systems, Inc.
Palo Alto, CA 94301

IQ 584593

Abstract

The goal of this research is to achieve very intelligent telerobotic controllers which are capable of receiving high-level commands from the human operator and implementing them in an adaptive manner in the object/task/manipulator workspace. ~~This paper presents~~ initiatives by the authors at Integrated Systems, Inc. to identify and develop the key technologies necessary to create such a flexible, highly programmable, telerobotic controller. The focus of the discussion is on the modeling of insertion tasks in three dimensions and nonlinear implicit force feedback control laws which incorporate tool/workspace constraints. Preliminary experiments with dual arm beam assembly in 2D are presented.

are presented.

I. Introduction

In the future, telerobotic manipulators will be used to enable increased space assembly, servicing, and repair. Specific goals, as determined by NASA, are:

1) "to decrease mission operations manpower by 75 percent,
2) to replace 50 percent of extravehicular activity (EVA) with telerobotics, and
3) to enable remote (e.g. geosynchronous earth orbit and polar orbit) assembly, servicing, and repair through telerobotics" [1].

In order to satisfy the above requirements for telerobotics, significant improvements in manipulator control will be necessary. Telerobotic assembly requires powerful locally autonomous control laws for 1)task completion in the presence of disturbances and sensor errors 2) control of position and force for trajectory guidance 3) task completion without continuous operator supervision. The last is especially important for long distance tasks (such as Mars exploration) where the time delays involved in receiving sensory information or relaying earth-based control signals make traditional teleoperation unsuitable. Furthermore, an interface for expert system planners and/or human interaction will be necessary so that the system is flexible to various levels of human supervision.

Previous work in the general area of robotics has focused on a decompositon of robot control into trajectory planning and servocontrol to the preplanned wire in statespace. A logical extension of this work approaches the problem with real-time expert systems to formulate the planned trajectory "in-the-loop". Expert systems "in-the-loop" will be much more powerful with more sophisticated control algorithms as a foundation. Namely, analytic, optimization based, "trajectory feedback", nonlinear control laws whose performance index and time phasing are controlled by a combination of expert systems and the human operator.

The effort described below involves fundamental nonlinear control laws valuable in dual arm coordination. These approaches to dextrous, coordinated motion were evaluated in a new highly flexible simulation environment [2]. The authors have modeled two 3DOF planar robots, performing a beam assembly task. Two dimensonal plots and figures illustrate by comparative results the sensitivity of performance to the control law structure. Reasonable long term research conclusions will be

a) which control approaches are most reasonable

b) what level of actuator/sensor performance is required to do meaningful experiments well with these control laws, and

† Manager, Aircraft and Robotics Control Systems Division
* Members of Technical Staff

c) what are the controller architecture issues to implement such control laws.

We first describe the modeling, then various control structures followed by a discussion of the experimental results.

II. Dual Arm Modeling

Modeling and Simulation Tools

The following research was carried out using MATRIX$_X$/AR (Automation and Robotics Modeling and Simulation Package)[2]. This tool provides a flexible modeling and simulation environment for manipulators, actuators, and control laws. Figure 1 is a flowchart for the use of MATRIX$_X$/AR. Model creation is initiated by using the menu-driven RUI (Robotic User Interface) to specify the geometric, inertial, and functional specifications of the manipulator. The RUI creates a robotic database from the given and computed data. This data is accessed by command files which build a kinematic and dynamic model of the manipulator using the recursive Newton-Euler approach. This model is created in SYSTEM.BUILD[3], a simulation/integration package where linear, nonlinear, and multirate systems and controllers can be modeled quickly and easily in a block diagram format. Figure 2 shows the inverse dynamic model of the PUMA 560 with blocks for the base, arm, wrist, and end effector. The blocks are nested, so that the block for the arm contains blocks for link 1, link 2, and link 3, each of which contains a dynamic, kinematic, and actuator block. By using input flags, the PUMA block can be used to obtain kinematic, dynamic, and inverse dynamic information. Suitable control laws can be generated by using classical and modern control design techniques available in MATRIX$_X$/AR. The plant and control models are combined in an overall system which is then simulated. Post-processing animation capabilities are used to visualize the success or failure of a particular controller.

Manipulator Robot Models

Each of the two robotic manipulators modeled in this study is a three DOF articulated arm. The arms are made up of three rigid links connected through one prismatic and two revolute joints. Since the first and second joint axes are perpendicular, and the second and third are parallel, each arm moves in a plane with one translational and two rotational degrees-of-freedom. A schematic of an arm is shown in Figure 3.

The 3-DOF planar manipulators are identical, with the physical characteristics shown in Table 1.

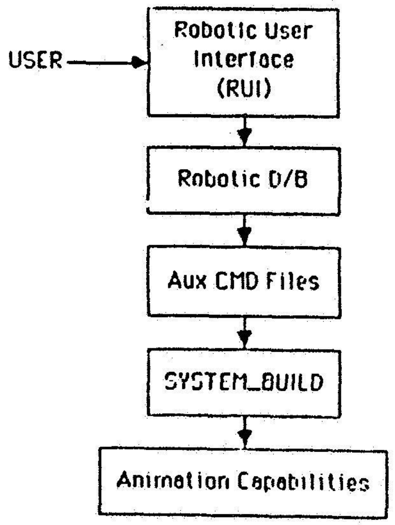


Figure 1. MATRIX$_X$/AR Design Flowchart.

Space Assembly Beam Model

The mating of two long slender beams was chosen as a suitable test for the proposed control algorithms. The beams were sized relative to the arms to simulate a truss assembly scenario. The beams, as well as the manipulators, have a

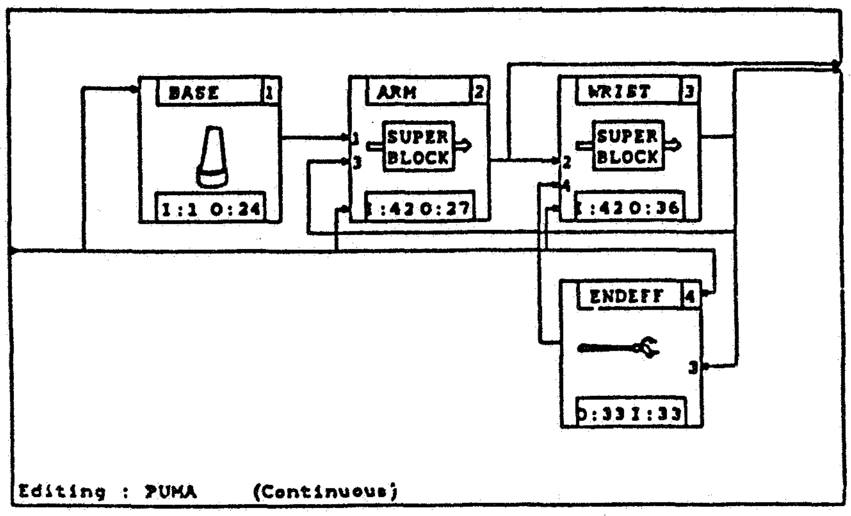


Figure 2. Model of PUMA 560.

cylindrical (hollow) shape with the dimensions shown in Table 1.

The dual arm configuration, complete with beams, is shown in its initial position in Figure 4. The arm and beam on the left with the socket will be referred to as Manipulator 1, and those on the right with the peg as Manipulator 2. The

Table 1
3-DOF Planar Manipulator Physical Characteristics.

	Link 1	Link 2	Link 3	Beam
Geometric Properties				
Joint Type	Swivel	Sliding	Hinge	n/a
Length (m)	.2	.4	.4	1
Cylinder:				
Inner Radius (m)	.046	.0335	.0335	.023
Outer Radius (m)	.050	.0375	.0375	.025
Inertial Properties				
Mass (Kg)	.724	1.07	1.07	.905
Inertias (Kg-m^2):				
I_{xx}	.0017	.0014	.0014	.0005
I_{yy}	.0032	.0149	.0149	.0757
I_{zz}	.0032	.0149	.0149	.0757

desired goal involves inserting a peg on the left end of the second beam into the hole in the middle of the first beam.

Dual Arm Constraint Model

Simulating closed dynamic chains, such as the dual arm manipulators during an assembly task, is a difficult problem. The collisions which occur during insertion result in abrupt changes in the motion (velocity) states. These discontinuities cause problems for the integration package. The problem is dealt with in this research by using a compliant model, since there is, naturally, compliance in any mechanical mechanism. For this work, the first arm, second arm, and second beam

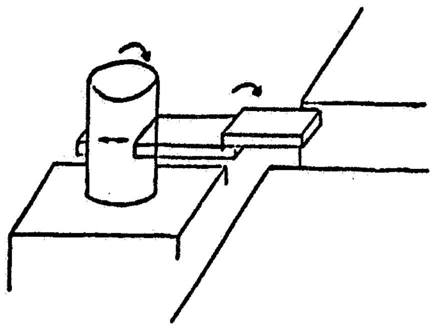


Figure 3. Schematic of Three DOF Arm.

are treated as being rigid. The hole and the second beam are compliant, generating an opposing force linearly proportional to the amount of deflection caused by the inserting peg upon collision. Since computer CPU time is dependent upon the "stiffness" of the equations, preliminary tests are run using relatively low spring constants. The results described below are in this category, with spring gains of 1000 N/m.

III. Dual Arm Control

This section gives an overview of the various aspects of the control approaches investigated on a dual arm experiment.

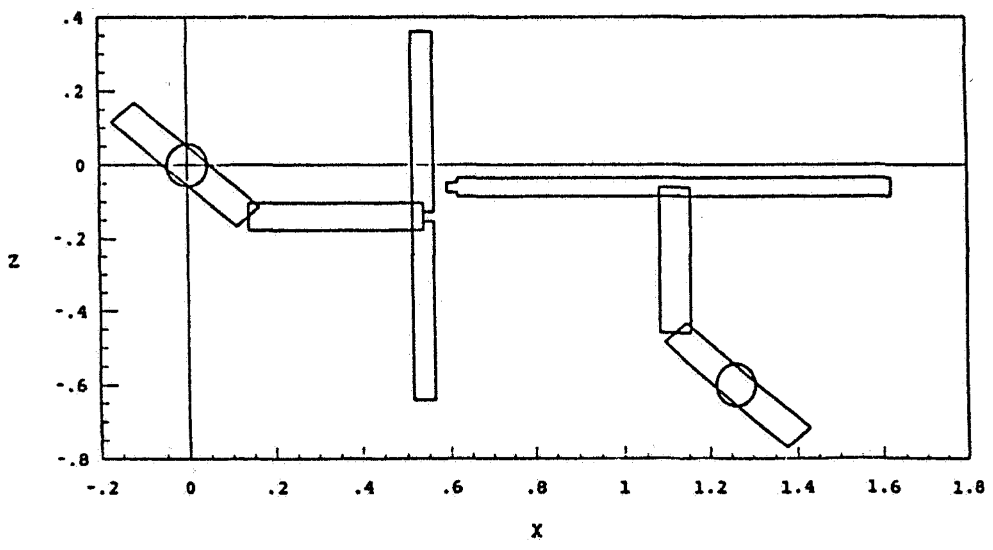


Figure 4. Dual Arm Configuration.

3.1 Control Design

The performance of a robotic manipulator in a compliant task, such as peg insertion, greatly depends on the choice of control used. A one-step control law based on previous research [4] was used for the beam assembly problem described above.

The advantage of this scheme over typical hybrid force/position control is that the control law works for both constrained and unconstrained motion; thus, the peg does not need to be in close proximity to the hole initially. A brief description of the control law is presented below.

Nonlinear Control

The equation of the motion of the end effector in cartesian space is given by

$$\Lambda(x)\ddot{x} + \mu(x,\dot{x}) + p(x) = F \tag{1}$$

where x is a vector of position and orientation, $\mu(x,\dot{x})$ contains the Coriolis and centripetal terms, and $p(x)$ contains the gravity terms. A nonlinear control law can be used to globally linearize equation (1) [5]. The result is a linear system of the form

$$\ddot{x} = F_c. \tag{2}$$

Multiplying (2) on both sides by $\Lambda(x)$ and adding $\mu(x,\dot{x}) + p(x)$ gives

$$\Lambda(x)\ddot{x} + \mu(x,\dot{x}) + p(x) = \Lambda(x)F_c + \mu(x,\dot{x}) + p(x)$$

Letting $F = \Lambda(x)F_c + \mu(x,\dot{x}) + p(\dot{x}) = \Lambda(x)F_c + F_d$, where $F_d = \mu(x,\dot{x}) + p(x)$, the feedback is composed of two components: a component containing the feedback law designed for the linear system (2), and a nonlinear decoupling component based on the nonlinear terms of (1). It is important to note that exact nonlinear control requires a precise model of the manipulator.

Constrained Motion Control

Control in the presence of constraints was based on research done by Ish- Shalom [6]. The method involves specifying a task constraint and then using that constraint as the optimization criterion for a linear quadratic optimal control design. For example, the constraint on the end effector force and velocity

$$f \cdot v = 0$$

describes sliding along a surface. A linear quadratic controller can be designed to satisfy this constraint. It is based on *minimizing the performance index*

$$J = ||f \cdot v||^2 = v^T H(f) v$$

and is derived for system (2) with the following result [4]:

$$F_c = -Kx$$

$$\begin{aligned} K &= [K_p \quad K_v] \\ &= [0 \quad \tfrac{1}{\sqrt{\alpha}} H(f)] \end{aligned}$$

where K_p is positional feedback, K_v is velocity feedback, α is a constant, and

$$H(f) = \begin{bmatrix} f_x^2 & f_x f_y & f_x f_z \\ f_x f_y & f_y^2 & f_y f_z \\ f_x f_z & f_y f_z & f_z^2 \end{bmatrix} \geq 0 \qquad \forall f \in R^3$$

Note that the force is controlled implicitly through the velocity feedback.

Unconstrained Motion Control

Control in the absence of constraints can be determined by using linear quadratic optimal control. The solution for system (2) will provide position and orientation control for the manipulator.

3.2 Dual Arm Beam Assembly Control Strategies

The three control laws described above were combined and used with the dual arm configuration described in chapter 2. Three preliminary control strategies, shown in Table 2, were chosen to determine preliminary conclusions on the importance of implicit force control and relative vs. global cooperative control schemes.

Table 2
Dual Arm Control Strategies Simulated

Ex.	**Control**	**Strategies**
#	***arm 1 (Socket Receptor)***	***arm 2 (Insertion Peg)***
1	NL global servo	NL-force global servo
2	NL-force global servo	NL-force global servo
3	NL-force global/ local servo	NL-force global servo

A global servo means the control is servoing to a point in inertial space. A local servo means that one arm servos relative to the other arm. The nonlinear aspect is what is often called the coupled torque method, and the force control is all implicit based on the constraint modeling described above. The next section describes the motivation for these strategies and experimental simulation results.

IV. Dual Arm Simulation Results

The first experiment tests the performance of control laws without coordination. The second experiment adds implicit force control to give a local coordination effect, and the third shows how significant passing information on the other arm's activity is to accomplish a coordinated assembly task.

Experiment 1 Objectives and Results

In experiment 1 both arms were servoed to a globally defined position and orientation. The defined position for both arms corresponds to the base point of the hole in the beam attatched to arm 1. Arm 2 was controlled by the combined controller as described above. Arm 1 however, did not have implicit force control. The simulation results are shown for successive time frames in Figure 5. Note that the initial contact of arm 2 onto arm 1 caused significant deflection, so that mating was only possible after a second attempt.

Experiment 2 Objectives and Results

Experiment 2 was the same as the experiment 1 with the exception that arm 1 was given implicit force control. As can be seen in Figure 6, the mating was accomplished in the first try. This is due to the cooperative motion of the manipulators after contact, even though they had no information about each others positions and were servoed to a globally defined point in space. The presence of the implicit force in arm 1 caused that arm to move in the positive z direction after being hit by arm 2 (perpendicular to the direction of the external force) rather than in the -x direction as before. Thus, local relative movement (away from the defined servo point) occured with arm 1 which allowed the two beams to mate faster and then travel back together to the global servo point.

Experiment 3 Objectives and Results

Experiment 3 was the same as experiment 2 with the exception that arm 1 was given information about the z component of the peg's location (attatched to arm 2). Arm 1 was thus servoed globally in the x direction and relatively (to arm 2) in the z direction. As can be seen in Figure 7, faster mating was obtained due to global movement of arm 1.

Simulation Experiment Summary and Conclusions

It should be noted that the above experiments represent preliminary results run under idealized conditions. The nonlinear control was exact and the hole was modeled compliantly as a spring system, allowing the manipulator to penetrate the first beams surface and then apply a point force proportional to the maximum penetration. These simulations were designed, however, to illustrate the potential benefit of using nonlinear implicit force feedback. Two key observations can be made.

First, the presence of implicit force feedback in both arms demonstrated how the two arms were capable of moving cooperatively without any knowledge of each other. The implicit force feedback allowed local movement about the globally defined servo point, which resulted in cooperative relative movement for the arms. This is extremely valuable since this can compensate for sensor inaccuracies in specifying and measuring the global servo point.

Second, as should be obvious, giving one or both arms information about each other, such as positional information, allows a greater degree of cooperation in the assembly task. Thus, future research will concentrate on using cooperative schemes, such as dual arm one-sided optimal guidance, to increase the amount of cooperation between the two arms.

V. Summary

This paper has outlined telerobotic research in progress at Integrated Systems. The emphasis on the work has been to develop goal directed guidance laws which provide a more powerful framework for telerobotic planners and teleoperator controllers to interface. Preliminary work has been done to test the concepts by simulation, using flexible automation modeling and control tools developed at Integrated Systems.

The dual arm control laws tested show that the control strategy is very important for assembly operations and could be of great benefit to NASA's space bound manipulators. Since there is a major need for telerobots to possess significant decision-making capabilities before they can be used extensively in remote and hazardous situations, it will be valuable in the industrial and nuclear environments as well.

References

[1] M.D. Montemerlo, "NASA's Automation and Robotics Technology Development Program," *Proc. of 1986 IEEE International Conference on Robotics and Automation*, San Francisco, CA, April 7–10, 1986.

[2] Reza Langari, Robert A. Walker, Norm Coleman, and Pak Yip, "A Robotics Modeling and Control Law Development Environment," *Proceedings of the AIAA Guidance Control Conference*, Williamsburg, VA, August 18–20 1986.

[3] N.K. Gupta, D. Varvell, and R.A. Walker, "SYSTEM_BUILD: A New Interactive Model Building and Simulation CAE Tool," *Proc. of the Society for Computer Simulation*, July 22–24, 1985.

[4] Reza Langari and Robert A. Walker, "Nonlinear Feedback Strategies for Control of Robotic Manipulators," ISI Technical Memorandum 6507–002, March 1986.

[5] E. Freund, "Fast Nonlinear Control with Arbitrary Pole Placement for Industrial Robots and Manipulators," *Robotics Research 1*, MIT Press 1982.

[6] J. Ish-Shalom, "The C.S. Language Concept: A New Approach to Robot Motion Design," International Journal of Robotics Research, Spring 1985, pp. 42–58.

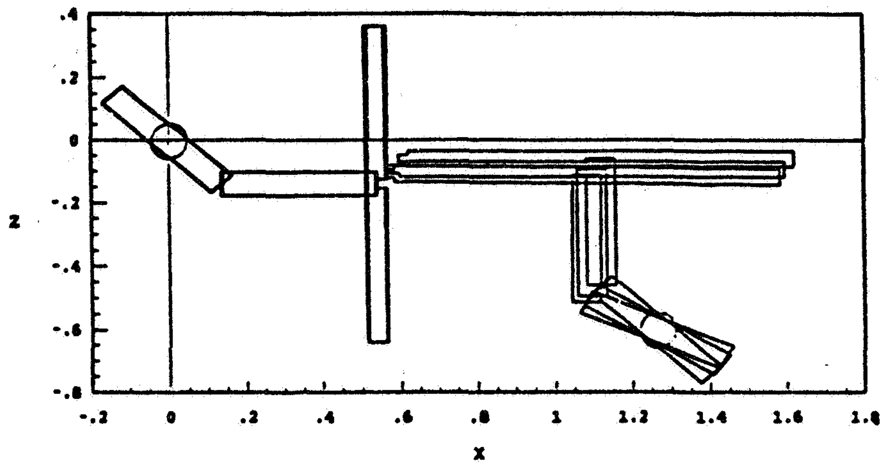


Figure 5a).

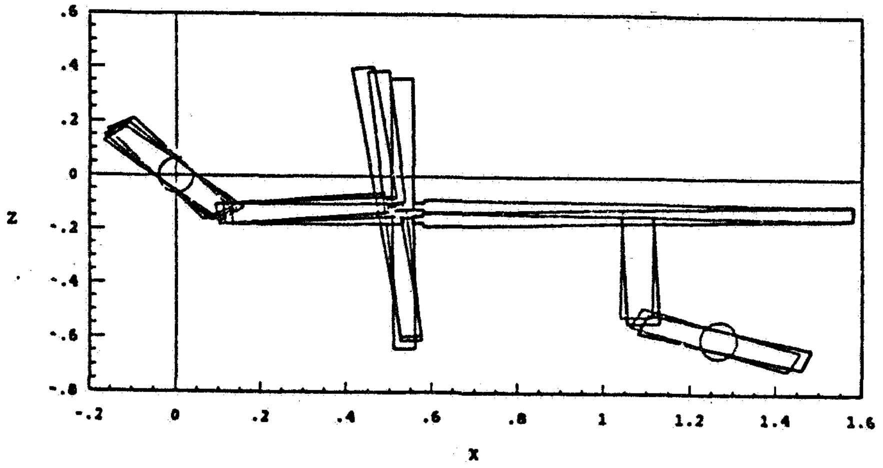


Figure 5b).

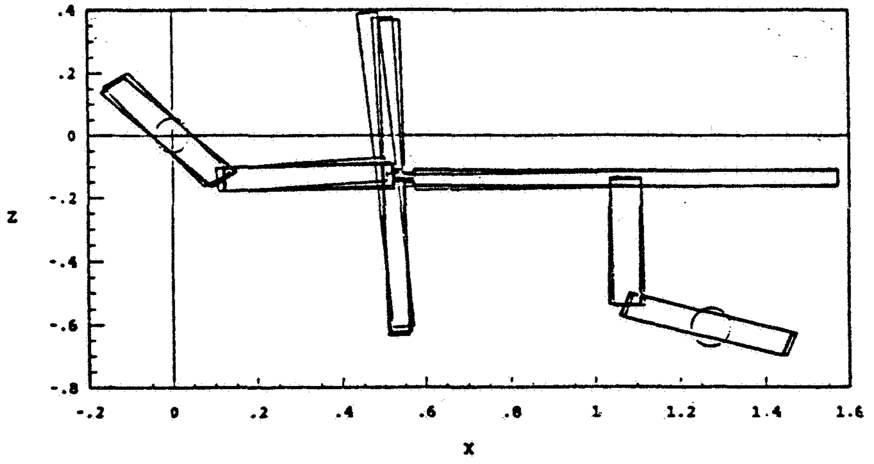


Figure 5c).

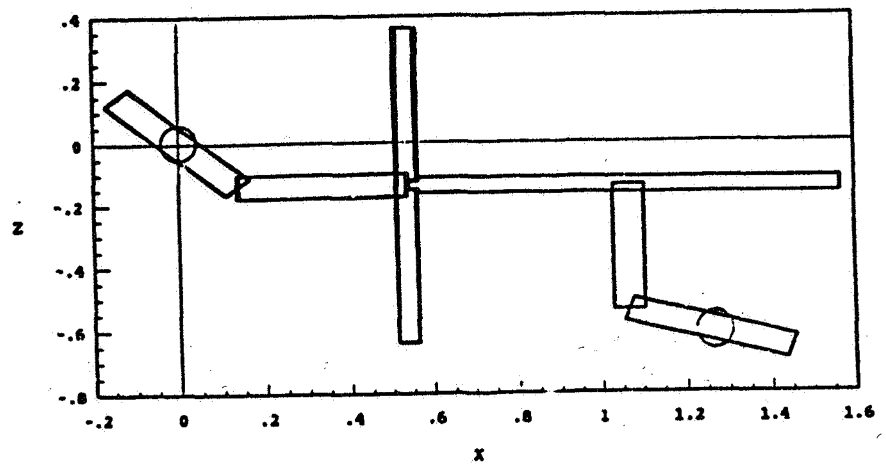


Figure 5d).

Figure 5). **Experiment 1 Results - Global Servoing With One-Sided Implicit Force Coordination (see Table 2) a) 0-1.75 secs, b) 2.25-3.25 secs, c) 3.75-6.0 secs, d) 6.75-9 secs.**

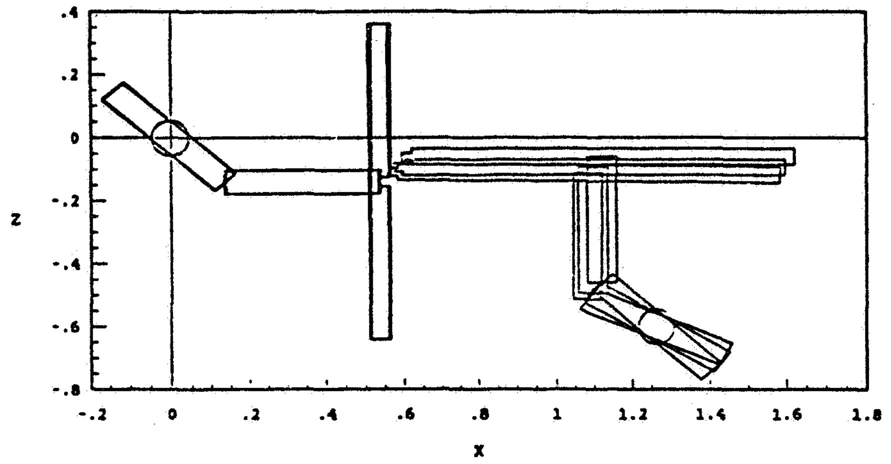


Figure 6a).

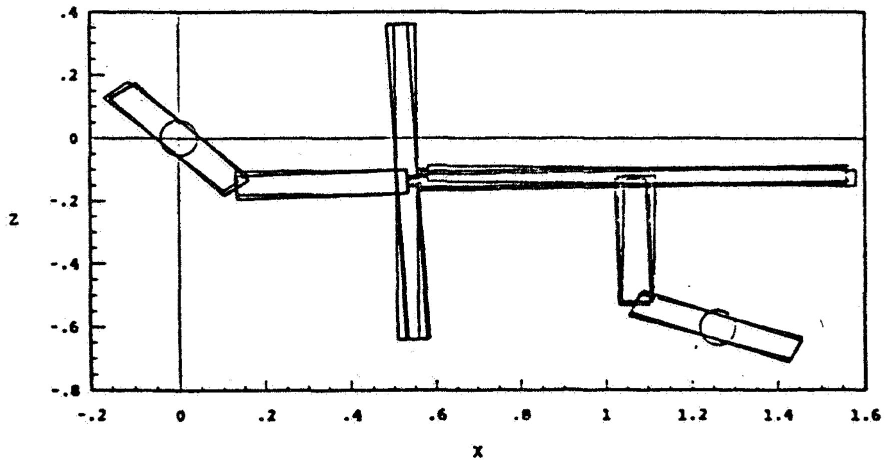


Figure 6b).

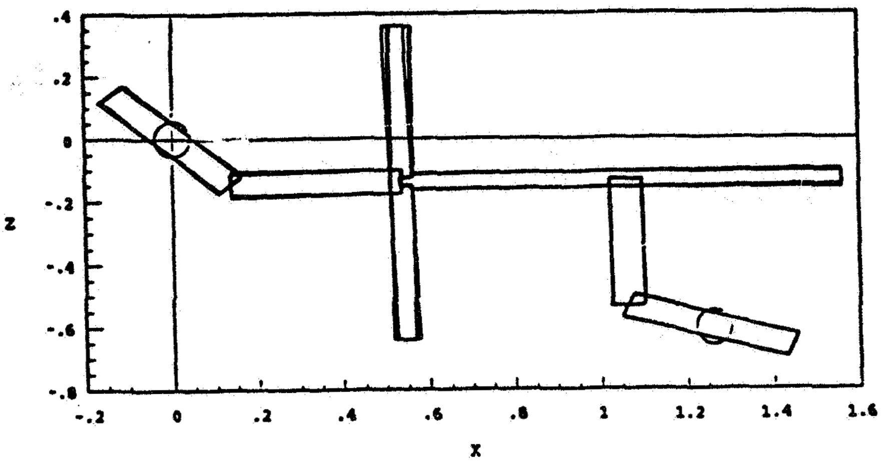


Figure 6c).

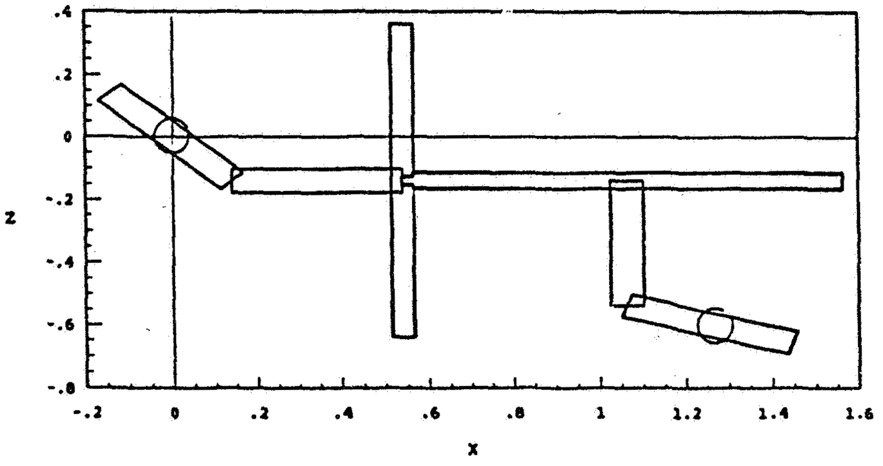


Figure 6d).

Figure 6). **Experiment 2 Results - Global Servoing With Two-Sided Implicit Force Coordination (see Table 2) a) 0-1.75 secs, b) 2.25-3.25 secs, c) 3.75-6.0 secs, d) 6.75-9 secs.**

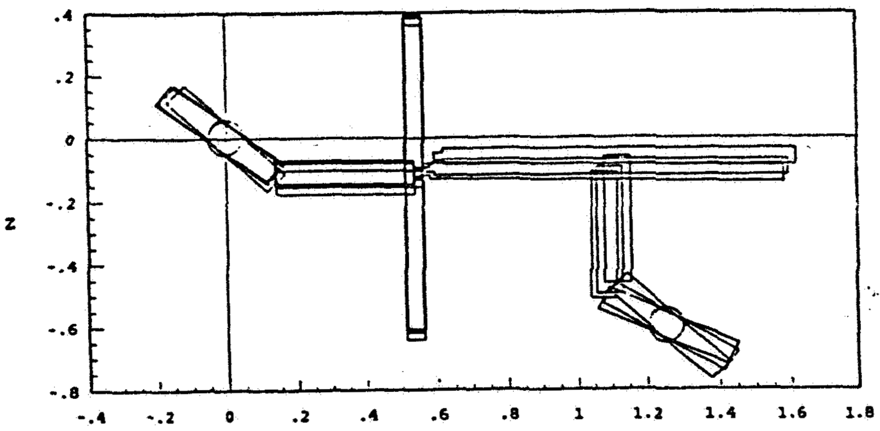


Figure 7a).

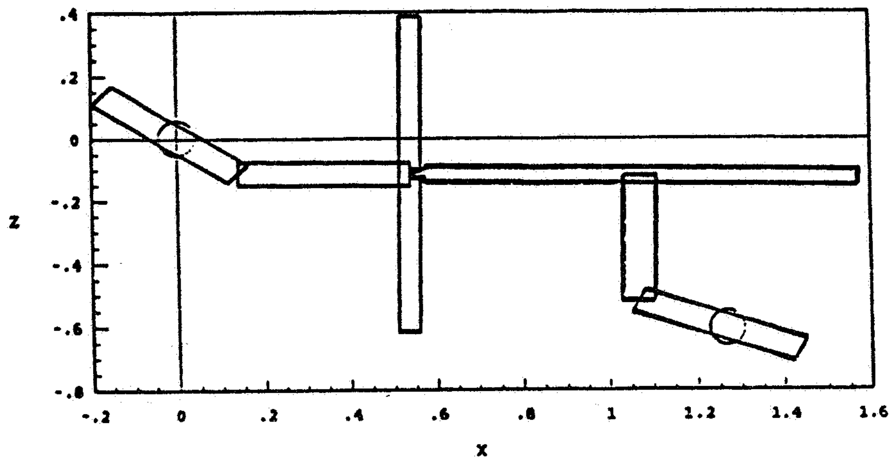


Figure 7b).

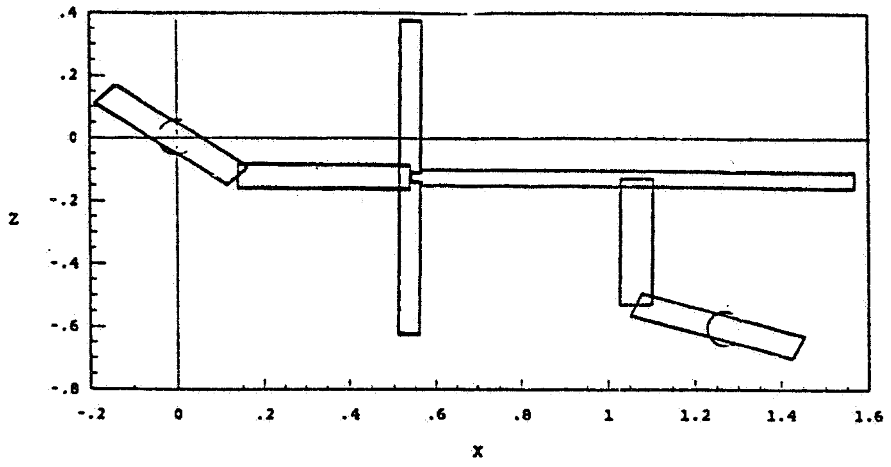


Figure 7c).

Figure 7). **Experiment 3 Results - Global and Local Servoing With Two-Sided Implicit Force Coordination (see Table 2)**

a) 0-1.75 secs, b) 2.25-3.25 secs, c) 3.75-6.0 secs.

S23-37
10P.
188143

A Universal Six-Joint Robot Controller

D.G. Bihn
Hewlett-Packard
Cupertino, CA 95104

T.C. Hsia
University of California, Davis
Davis, CA 95616

HR 501144

CC 907079

1. Abstract

A general purpose six-axis robotic manipulator controller was designed and implemented to serve as a research tool for the investigation of the practical and theoretical aspects of various control strategies in robotics. A 80286-based Intel System 310 running the Xenix operating servo software as well as the higher level software (e.g. kinematics and path planning). A Multibus compatible interface board was designed and constructed to handle I/O signals from the robot manipulator's joint motors.

From the design point of view, the universal controller is capable of driving robot manipulators equipped with D.C. joint motors and position optical encoders. To test its functionality, the controller is connected to the joint motor D.C. power amplifier of a PUMA 560 arm bypassing completely the manufacturer-supplied Unimation controller. A controller algorithm consisting of local PD control laws was written and installed into the Xenix operating system. Additional software drivers were implemented to allow application programs access to the interface board. All software was written in the C language.

2. Introduction

Robots are becoming increasingly prevalent in the industrial workplace, as well as creating an industry of their own. This new industry is both driving and being driven by new technologies. New materials, improved mechanical designs, and faster controller electronics are running into the limitations of traditional control techniques. Thus, theoretical work to overcome these limitations is urgently needed. Much of the theoretical work is being carried out in academic research institutions. However, there is often a significant gap between the results of theoretical studies based on simulations and the verification based on actual implementation. Industry is often reticent to try untested theoretical results, preferring the time-tested, sub-optimal control techniques of the past, possibly sacrificing substantial performance improvements. A credible testing ground for new control techniques is needed to bridge the gap between theory and application.

The Robotics Research Laboratory at the University of California, Davis, has a Unimation PUMA 560 robot arm representative of a large and popular class of modern industrial manipulators. The PUMA arm is controlled through the sophisticated robot language, VAL-II. The user only has access to the arm through high level 'move-type' commands. He therefore has little control of the actual arm trajectory and no control over the low level motor servo loops. In typical industrial applications, the inability to alter low level functions of the controller does not represent a functional limitation. To the contrary, it actually affords both the arm and the operator a fair degree of protection and safety. The academic researcher, however, is prevented from using the arm to test and demonstrate new control strategies and is forced to rely on computer simulation.

3. Objective

The objective of this project was to design and implement a computer based robotic controller which allows the researcher to write programs and implement algorithms which control the robot arm from the lowest level of the closed-loop servo system to the higher levels of kinematics, dynamics, path planning and robot language [11]. The use of a familiar software environment was chosen with the intent of making the user interface as clean and simple as possible.

The scope of this project is limited to the design and implementation of a controller consisting of (1) the Joint Interface Board electronics, and (2) the operating system interface to this hardware. A simple low level 6-joint P.I.D. (Proportional Integral Derivative) controller is implemented and presented to serve as both a functional test of the system and as an application example. The topics of joint kinematics and other high level application software are beyond the scope of this project as is the advanced control law design.

4. The Controller System

The contro:ler presented here is designed around an Intel 310, 80286-based, microcomputer [2] running the UNIX-like operating system, XENIX [3]. A signal interface board was designed and constructed to provide the interface between the microcomputer and the joint motors of the arm. The Unimation controller, supplied as part of the PUMA 560, was modified to serve two low level functions: as a convenient access point for the joint feedback signals from the arm and as a multi-channel power amplifier drive the joint motors. All other electronics in the Unimation multi-channel power amplifier to drive the joint motors. All other electronics in the Unimation controller are by-passed; closed-loop control is done in the Intel-based controller described here. The controller system is depicted by the block diagram shown in Figure 1.

A single 80286 CPU running at 6 MHz is used to execute both high level (e.g. kinematics) and low level (e.g. joint servo loops) control software. At a typical sampling rate of 100 Hz, about 30% of the CPU time is required to execute the six P.I.D. controllers implemented in the design example. The remaining CPU time is available for application programs and the operating system. The interface board itself is useful in systems with sampling rates over 2 KHz. However, to utilize this speed, additional CPU power is required.

5. System Design Requirements

Two basic elements constitute the controller system designed and implemented in this project: a digital computer and special purpose interface hardware. The digital computer performs all the control functions, from the joint motor servo control law to the higher levels of coordinated joint motion. The interface hardware function is to provide the basic link between the computer and the physical signals required to control the robot arm.

5.1 The D.C. Servo Motor Position Measurement

The control of the robot arm is equivalant to the control of the joint motors. In this controller, D.C. servo motors are assumed to be equipped with potentiometer and/or incremental encoder position feedback devices. It is also assumed that the D.C. motor can be driven by an analog (voltage) signal buffered by an appropriate external power amplifier (servo motor amplifier). The Unimation PUMA 560 arm has six geared D.C. servo motors with both encoder and potentiomenter position feedback elements and it is considered to be prototypical of the class of manipulators considered in this project.

Each motor, in general, does not directly drive a manipulator joint, but is typically connected through a gear train requiring a multiple number of motor revolutions to drive the joint through its operating range as shown in Figure 2. In the configuration assumed in this project, feedback elements are directly attached to the motor, not the actual joint member. Joint position is inferred from the motor position. This requires that absolute motor position must be measured over multiple revolutions. In the PUMA arm, both a geared (i.e. multi-turn) potentiometer and an incremental shaft encoder are connected to the motor shaft to collectively supply this data. The incremental encoder is used to accurately measure both the relative motor position over an arbitrary number of rotations and the absolute motor position modulo one rotation. The geared potentiometer is used to measure the approximate absolute motor angle over the several revolutions needed to drive the joint through its range. Once the absolute motor angle has been determined, only the relative data supplied by the encoder is needed.

The incremental encoder, which is directly attached to the motor, generates two types of data: (1) high resolution quadrature signals which are decoded into relative (incremental) angular displacement information and (2) an index pulse which is produced once per revolution and can be used to accurately define the absolute angular position of the motor modulo 360° (Figure 2).

The geared potentiometer supplies indirect, low resolution absolute joint position data. The gear ratio of the potentiometer is designed so that when the joint is driven between its mechanical limits, the pot wiper rotates within its mechanical limits (less than 360°). Logically, this pot could have been attached directly to the robot joint. For manufacturability considerations, the pot has been included in the motor assembly.

Once the absolute motor position has been determined, it is continuously updated (incremented or decremented) by the data from the incremental shaft encoders. As long as the electronics are not interrupted (e.g. power-down) the data from both the geared pot and the encoder's index pulse are not used. The difficulty is the initial determination of the absolute motor position is rather involved, and will now be discussed.

When the absolute motor position is unknown, the potentiometer wiper voltage can be measured and the absolute motor position estimated. Once estimated, further position measurement can be made by monitoring the relative position data from the incremental encoders. While the incremental data is very accurate, the absolute position can only be as good as the initial estimate. The standard technique to obtain an accurate measurement of the initial absolute position is as follows. First, the motor is driven until the encoder's index pulse is found. At this point the absolute position is known to be an exact multiple of 360°. Next, the potentiometer voltage is measured to give the approximate absolute position. Combining the approximate absolute position with the certain knowledge that the position is an exact multiple of 360°, the exact absolute can then be derived.

The above explanation serves to demonstrate the basic idea and what sort of precision is required. For analysis, the actual parameters of the PUMA 560 joint motors are used to determine the system specifications and Joint Interface Board requirements.

5.2 A Typical D.C. Servo Motor

The PUMA 560 servo motors are integral packages which contain four basic components: (1) a D.C. motor; (2) an electric brake; (3) an optical incremental encoder; and (4) a geared-down potentiometer. The currents activating the motor and the electric brake are the inputs while the encoder and the potentiometer signals are the outputs. The basic functions needed to operate the motor system are described below.

5.2.1 Reading the Incremental Encoder

The incremental encoder has three output signals: channels A, B, and the index pulse. Channels A and B are used to determine both the amount and direction of rotation in discrete steps. The index pulse produces a single short pulse each motor revolution which can be used by the system to determine the absolute angle of the motor and, with the addition of the potentiometer data, can be used to determine absolute position (described above).

The output states of channels A and B are used to detect relative motion (rotation) of the motor shaft and in turn, the joint itself. How this is done is well-known and is not described here.

5.2.2 Counting the State Changes

The incremental encoders on the PUMA 560's motors produces 1000 state transitions per revolution, except for the shoulder joint (#2) which produces 800 transitions. The motor (with the encoder directly attached) rotates from 40 to 60 times during full joint travel (depending on the joint), corresponding to 40,000 to 60,000 state changes per complete joint motion. It is convenient if the hardware keeps count of the total joint range. This way the total joint motion may be read directly from the hardware counters. 16-bit counters have a maximum count of 65,532 and are sufficient to keep track of the joint motors of the PUMA arm. However, it is not essential for the hardware to count the entire joint range. In a sampled data system, the software can keep track of total joint motion, using the hardware only to count the relative motion which has occurred between samples. If the hardware count is used in this manner, the absolute motion is limited only by software and the incremental motion between samples is limited to motion of ± 32K pulses.

5.2.3 Reading the Potentiometers

The potentiometers incorporated into the PUMA 560 joint motors are connected between +5 volts and ground. Rotating the pot through 360° produces a proportional voltage output from 0 to 5 volts (e.g., 90° produces 1.25 volts, 180° produces 2.5 volts, etc.). These pots have been geared so that they rotate somewhat less than 360° for a complete joint movement; depending on the joint, full joint travel may produce as little as 200° of potentiometer motion. This restricted travel corresponds to a change in pot voltage of about 2.78 volts. If the joint produces 60 index pulses (i.e. 60 motor rotations) per full joint motion, the pot voltage must be measured to an absolute accuracy of 1/60th of 2.78 volts (0.046 volts) in order to determine the motor shaft angle to within one revolution.

An Analog to Digital Converter (ADC) is used to measure the pot voltages. It must be able to measure a voltage which spans a 0 to 5 volt range, and must have a resolution and accuracy of better than 0.046 volts over this range. This corresponds to a full-scale resolution of 0.92%. A seven-bit ADC has a resolution of 0.78% and is sufficient for this voltage measurement.

Since the potentiometers are not part of the dynamic control scheme presented here, there is no constraint on the conversion speed. For the PUMA arm, both speed and resolution requirements of the ADC are easy to meet. However, to make the system more flexible, other possible applications should be considered. It is often the case where a symmetrical voltage signal (say -5 to +5 volts) needs to be measured and fast conversion time can make dynamic control systems with analog feedback elements possible. Furthermore, since fast (30 microsecond) 12-bit ADCs with input range of ±5 volts are conveniently available and at reasonable cost, this higher performance device was chosen.

5.2.4 Driving the Motor

The drive current and voltage needed by a D.C. motor depends on the size and type of motor used; no solution is appropriate for all motors. It is therefore considered impractical to include the power amplifier as part of the design. The important requirement is how to drive these power amplifiers.

In general, two standard techniques for supplying the current needed for driving D.C. servo motors are commonly used: linear amplifiers and pulse width modulated (PWM) amplifiers. Each have advantages but the important fact to consider is that they both are controlled by a simple analog voltage.

In the particular case of the PUMA 560 arm, the Unimation PUMA controller's power amplifiers can be conveniently used because they have been designed explicitly to drive the PUMA 560 joint motors. Using this controller also makes the external connection to the arm joint motors simple and straightforward. Additionally, the Unimation amplifier has several useful safeguards which automatically shut the amplifier off to prevent damage to the arm.

Power amplifiers are controlled by analog voltages, and to generate these voltage outputs from a digital controller a Digital to Analog Converter (DAC) must be used. Three basic specifications must be considered: (1) voltage swing; (2) resolution (number of bits); and (3) the accuracy. Commercially available power amplifiers typically require a voltage input of -10 volts to +10 volts. This also corresponds to typical DAC device output characteristics, and the input specifications of the Unimation controller's power amplifier.

Selection of resolution and accuracy is more difficult. 8-bit corresponds to a resolution of one part in 256 (0.39%), 10-bit corresponds to one part in 1024 (0.098%), and 12-bit corresponds to one part in 4096 (0.024%). A 10-bit unit was chosen and considered to be a reasonable compromise between price and performance.

5.2.5 Releasing the Brake

The brake is used to lock each joint in position when the servo motor is turned off and is necessary to keep the arm from collapsing. The brake is much like a D.C. relay. When D.C. current passes through the coil (an electromagnet), the brake plate is retracted from the friction plate allowing the motor to rotate. When no current is flowing in the brake coil, the brake plate is forced into contact with the friction plate by compression springs and the motor cannot rotate. On the Unimate controller, the brake release current is supplied whenever the 'arm power' is on. This is a fail-safe system. When the servo power (to the motors) is turned on, the brakes are released. When the arm power is switched off, the brakes are automatically applied, holding the joint in place.

5.2.6 Other I/O Requirements

The Joint Interface Board must not only accommodate the joint motor signals but must also provide the host computer with additional functions to allow all subsystems to be integrated into a complete controller. Included in the design are (1) a digital timer and associated interrupt circuitry, and (2) 24 bits of general purpose I/O lines.

5.3 Host Computer Requirements

The selection of a suitable host computer is very important. The machine must not only be capable of meeting basic execution speed and I/O requirements, but should also be able to support the software tools needed to implement a controller. In this section both the host computer hardware and software are discussed.

5.3.1 The Computer: Intel System 310

The Intel System 310 microcomputer was used because it satisfies the above criteria. It is based on the Intel 80286 16-bit microprocessor [4]; the system also comes equipped with an 80287 floating point math co-processor [5]. It is a Multibus based system [6], a bus standard which is particularly popular in the area of industrial automation. A wide variety of interface board products, including memory, I/O, and blank proto-type boards, are available from Intel and third party vendors. A standard Multibus board is comparatively large which allows complex circuits to fit onto a single board, allowing the use of a single bus interface circuit. All the hardware for the Joint Interface Board was able to fit on a single board.

5.3.2 The Operating System Choice: XENIX 286

The Intel 310 can run several operating systems: the ubiquitous IBM PC's MS-DOS, the UNIX-like XENIX system O.S., and the real-time, multi-tasking systems RMX-86 and RMX-286.

XENIX was chosen to be the operating system of this project's implementation. A substantial learning effort is required to become proficient with an unfamiliar operating system and new software tools. XENIX minimizes this obstacle; many researchers are familiar with UNIX and need little time to master XENIX. Those unacquainted with UNIX can be motivated to learn XENIX since this knowledge is useful on many other systems. This is a very important consideration on short term projects where learning a new operating system may require more time than the experiment itself.

The XENIX operating system is Microsoft's licensed version of UNIX III with some of the Berkeley Software Distribution (BSD) enhancements (e.g. 'vi' and the C-shell), and several of their own enhancements. It is a multi-user system. UNIX is a very powerful environment for developing software and is widely used in the academic and research communities. The disadvantage is that it was not designed for real-time applications. Details of the techniques used to construct a real-time controller for our purpose are given later.

6. The Design

This section details the design and implementation of the above specifications. The discussion is divided into three sections: (1) the hardware design of the Joint Interface Board (JIB); (2) the connection between the J.I.B. and the Unimate PUMA 560 controller; and (3) the software interface between the XENIX operating system and the JIB.

6.1 Joint Interface Board Design

The block diagram which outlines the J.I.B. hardware is shown in Figure 3. As seen from the computer side of the bus interface, the JIB is a small collection of I/O devices: six 16-bit encoder counters; an encoder reset circuit; two PIO (parallel input/output) devices; timer and the interrupts reset logic. One of the PIOs is used exclusively to interface to the ADC and DAC subsystem, and the other PIO is used for off-board digital expansion.

6.1.1 The Analog-Digital Subsystem

Communication and control signals for all seven DACs, the ADC and the analog multiplexer are [illegible] to one of the PIOs on the system. Only the PIO is in the Multibus address space and control of these devices must be made through the PIO. The primary reason for designing the system this way was bus speed considerations. The PIO, an 8255, can operate at the full 80286 bus speed while the ADCs and DACs are about twice as slow (450 ns vs 180 ns). Rather than slow the bus down on this board and degrade performance of the other onboard devices (e.g. the encoders), the ADCs and DACs are given their own private slow bus.

6.1.2 Analog Output: The DACs

Six analog voltage outputs are necessary to drive the basic joint servo motors. An additional analog voltage output was included to permit future expansion, possibly the control of a more sophisticated gripper. To produce these outputs, seven independent DACs (digital to analog converters) were used. The independent DAC approach offers the advantage of a very straightforward interface, improved accuracy and simpler circuit design.

As described in the Analysis section above, the analog outputs must be capable of delivering a voltage from -10 volts to +10 volts at a resolution of 10 bits (1 part in 1024) to properly drive the inputs of the servo motor power amplifier.

6.1.3 Analog Input: The ADCs

As described in the analysis section, each of the PUMA 560 joint motors has a potentiometer which produces an output from 0 to 5 volts and, to be useful in absolute position determination, these signals must be resolved to an 8-bit accuracy. Fast, high resolution analog to digital converters can be obtained at reasonable prices which exceed the basic specification but give the Joint Interface Board more power. Analog Devices' AD574 [7] is a popular example. It has a 12-bit resolution, a conversion speed of less than 30 micro-seconds, selectable input ranges of 0 to +10, 0 to +20, -5 to +5, and -10 to +10 volts, and a cost of less than $35. At this speed of conversion, one device is fast enough to convert all six joints' pot data in less than 0.2 milliseconds, a speed fast enough to allow the pots alone to be used as the primary feedback element in situations where it may be useful.

To use one ADC to convert several analog input signals requires the use of an analog selector or multiplexer. A typical analog multiplexer, the LF1308 has eight voltage inputs which are selected to one output. This output can then be converted by the ADC, one at a time. Like the DAC outputs, ADC outputs must also be latched. However, since the ADC output is digital, it may be easily stored inside the computer using software without using any special hardware.

6.1.4 Timer Subsystem

Generating a constant sampling interval requires an external clock source to interrupt the CPU and cause the control software to execute. The Multibus provides a 10 MHz clock requiring an onboard frequency divider logic. To allow convenient changing of the sampling rate, a divide-by-ten prescaler followed by a microprocessor compatible programmable timer was selected. An Intel I8254 triple 16-bit timer I.C. [8] was used. It features extensive programmability, high resolution (one part in 65K). In the divide-by-n mode it can be programmed to generate a square wave with a period from 2 microseconds to 65 milliseconds in 1-microsecond steps. This corresponds to a rate from 22 Hz to 500 KHz (though rates above 200 Hz are not usable in the present system). Timer #0 is used as the interrupt clock, leaving timers #1 and #2 available for future applications.

6.1.5 Digital I/O

To make the Joint Interface Board a more flexible and general purpose interface, an additional parallel input/output (PIO) I.C. was included in the design. All the 24 outputs from this device go directly off-board via the connector J12 and are not used by any of the onboard electronics.

6.1.6 The Encoder Subsystem

The JIB accepts six sets of incremental encoder signals. Each input set is used to control its own 16-bit counter, instructing it to count UP, count DOWN, do nothing, or RESET to zero. The encoder subsystem can be divided into three parts: (1) the basic up-down counter; (2) the decode logic; and (3) the reset logic.

A. The Counters

The 16-bit up-down counter is a straightforward cascading of four 4-bit up-down synchronous counter with three control inputs: clock enable (CE), up-down select (UD), and reset (R). The system clock is running continuously at 1 MKz.

To directly implement a state decoder, six decoders would have to be constructed. This would probably require six 16-pin DIP packages. These would probably have to be either bipolar PROM (programmable read only memory) or some type of PLA (programmable logic array). If the PROM approach is used, a 16 x 2 = 32-bit PROM would be sufficient. The total number of bits required by all six units in this scheme is 196 bits.

This new 12-bit vector has 4096 possible states, each of which must be decoded to generate a 6-bit output vector, $\underline{O}$, with the proper CE and UD signals for three counters.

For six counters, a total of 4096 x 6 x 2 = 48 K-bits is required. This is two orders of magnitude greater than the scheme where each counter has its own state decoder. The advantage of this bit wasteful approach is that all this decoding can be done using just two 8 K-byte EPROMs packaged in 28-pin DIPs. These memory I.C.s are inexpensive and EPROM programmers are typically found in microprocessor development laboratories.

B. The Encoder Reset System

An index pulse signal is generated every incremental encoder (servo motor) rotation. This signal is used to supply quasi absolute position information about the motor so that the motor revolutions (e.g. 0°, 360°, 720°, etc.) can be distinguished from one another. Typically these index signals are only used during initialization of the hardware and software after system power up. Once the system has been initialized, incremental information alone is sufficient to determine absolute position (provided no encoder state changes are missed).

The basic scheme of the reset/calibrate routine is to rotate each motor until the index pulse is found and then this position is defined to be the position zero. Conceivably, this could be done in software by continuously reading the index signal until it is detected. This would require the software to sample the signal fast enough so that the pulse is not missed when the motor is moving at some speed.

To overcome this limitation, a hardware scheme was devised which allows calibration of the system with the motor to be running at any speed within its operating limits. Each counter has a synchronous reset input. The index signal from the encoder could be connected to this input causing the particular counter to reset to zero whenever the index pulse occurs. However, since the motor typically rotates tens of times during the joint travel, some form of selectively gating the index signal on and off was required.

This circuit is asynchronously set or 'armed' via the ARM RESET signal. Once armed, the next occurrence of the index pulse generates a single reset pulse for the associated counter circuit. Once the reset pulse is issued, the circuit disarms itself so that further occurrences of the index pulse will not reset the counters. The software can monitor these signals to check if the reset circuit is armed or not and can thereby determine if the index pulse has occurred.

6.1.7 The Multi-Bus Interface

Up to this point, all the subsystems described here have been computer independent (except for the general requirement of a 16-bit bus). This allows easy conversion to many other 16-bit computers such as the IBM-AT. At this point the design becomes specific to the hardware of the host machine. The Intel 310 system is based on the Multibus. The Multibus supports direct addressability up to one megabyte through a 20-bit address and 8- and 16-bit data transfers at a rate of five million transfers per second (10 MB/sec). The Joint Interface Board has been designed as a simple slave and never controls the Multibus. The JIB only decodes the address lines and acts upon the command signals from the bus master.

6.2 The Unimation Interface

The following sections describe how the Joint Interface Board and the XENIX software interface running on the Intel 310 were connected to the Unimate PUMA 560 arm. Position feedback signals from the arm servo motors are sent to the JIB, and the JIB sends analog voltage outputs to the power amplifier, which in turn drives the servo motor in each joint.

The Unimate PUMA controller consists of an LSI-11/73, six 6503-based joint controller boards, several low level interface boards, and a six channel-high current power amplifier. The controller presented in this project makes use of only the power amplifiers and one of the feedback signal conditioning circuits. The LSI-11 and the six microprocessor joint controllers are completely bypassed.

To close the loop around the joint motors, the feedback signals from the PUMA 560 have to be connected to the Intel system and the output command voltages must be returned from the Intel to the power amplifiers in the Unimate controller. It was considered desirable to make the necessary modification to the Unimate controller in such a way that switching between the Intel controller and the internal Unimate controller systems is as simple and safe as possible.

Connecting the feedback signals from the Unimate Controller to the Joint Interface Board is accomplished by inserting a proto-typing card (from here on called the Unimate Interface Board) into one of the several available empty, unwired slots of the joint controller portion of the Unimate card cage [9]. This technique was selected for several reasons. All of the PUMA arm feedback signals enter the controller through connector J-30 and are hard-wired directly to the ARM CABLE CARD in the card cage. Here some basic signal conditioning is performed, power is supplied to the joint pots and encoders, and the encoder outputs are then buffered to produce clean logic levels. Since these functions are required and would have to be duplicated if this subsystem was not used, it was convenient to use the external hardware and obtain these signals after conditioning.

The only place these feedback signals are found is on the backplane of the PUMA joint controller's card-cage. One of the available slots was chosen and all the necessary connections were made only by adding wires to the backplane, bringing all the feedback signals to the selected slot. This has the attractive feature of not having to break or cut any Unimate connections, leaving the controller intact. When the Unimation Interface Card is removed from its slot, the system is electrically and logically in its original condition. The card which is inserted into this slot also contains an inverting line driver to buffer the encoder signal to drive the wires connecting it to the Intel/JIB system.

While the feedback signals can be sensed without breaking any connections in the Unimate controller, this is not possible for the motor current command signals. In general, either the Unimate or the Intel/JIB joint motor current command signals (the DAC outputs) can be used, since only one controller can be selected to drive any motor at any one time. Current command signals enter the power amplifier through connectors P73 and P74. These connectors are located on the top of the POWER AMP CONTROL card and are readily detached. This is the only point where these signals can be 'intercepted' and the Intel signals injected without permanently modifying the circuit (e.g. cutting wires). A small interface panel with the appropriate connectors was fabricated to allow the JIB and Unimate motor current command signals to be selectively sent to the Unimate's power amplifier on a joint-by-joint basis (which is very useful during system debugging).

6.3 The XENIX to the Joint Controller Board Interface

The Joint Interface Board is installed in the I/O space of the Intel 310 (distinct from the memory space) and like all other system hardware in XENIX, the user can only access it through system device drivers. Drivers for all the JIB devices have been written and installed into XENIX (see software listing in Appendix H [1]). Application programs access these devices through symbolic names (e.g. "dac_1", "adc_4", "timer_1", etc.). The device driver controls the details of the data format and of physically addressing the hardware transparent to the application program [10].

Properly written drivers protect the system from the application programs and make the user interface clean and simple. *A motor controller can be implemented entirely at the application level, individually access*ing the incremental encoders, DACs and the ADC through their respective device drivers. While this will work, much of the CPU time is consumed in operating system overhead. Each I/O request (e.g. read and write) takes substantially longer to execute than if the software is able to directly address the hardware (not permitted in XENIX). An alternative to implementing the controller at the application level is to place it in the XENIX kernel as a single logical device (e.g. rather than "dac_1" and "encd_1" devices, a single "pid_1" can be considered as the basic I/O unit). Code written at the kernel level has direct access to the I/O space and may read and write to the JIB without going through the operating system. This reduction of overhead can reduce execution time by about 50%.

6.4 Real-Time Issues

XENIX is not a real-time operating system; it does not guarantee when a particular application program will get executed. It is often said that XENIX (vis-a-vis UNIX) does not guarantee when an interrupt is serviced. This only refers to the application level, not the lowest level of interrupt handling. In the common application of a terminal handler, an interrupt is issued from the serial interface hardware (a UART) each time a new character is received from the terminal. The interrupt handling software then services the hardware, taking the new character and putting it into the terminal handler's buffer. This software only competes with other interrupt routines (e.g. other terminals) for CPU time. Non-interrupt level operating system software which processes the characters in the terminal handler's buffer must compete with the entire system (including other application programs) for CPU *time and it is here where XENIX cannot guarantee response time. This issue* is important in designing a real-time controller.

A XENIX based real-time controller may be constructed in two fundamentally different ways. Both methods require that an external timer interrupt the CPU at fixed intervals and that kernel level device driver be installed in the XENIX system to process this interrupt. In the first method, the interrupt handler of the driver responds to the timer interrupt by only setting a flag in the driver's memory. When the 'device' is read by the application software, the read part of the driver tests this memory flag. If the flag is set, it returns back to the application program. If the flag has not been set (i.e. the timer has not yet interrupted the CPU), the read routine keeps testing the flag until it is set by the timer interrupt. This technique allows application programs to synchronize *themselves to the external clock and produce a constant sampling rate for a* digital controller written at this level. However, XENIX does not guarantee when the application program will be allowed to execute, and this may lead to occasional missed sampling intervals. As long as XENIX is run in the single user mode and the timer interrupt is not faster than 100 Hz, a useful system can be implemented.

In the second method, the one used in this project, the entire control system software is installed at the kernel level of XENIX and is executed as part of the interrupt service routine of the driver itself. Since the interrupt service routine does not have to compete with the non-interrupt portion of XENIX (including all application programs), this technique is guaranteed to be executed on each timer interrupt, producing a reliable sampling interval.

This is an effective method of implementing a real-time controller in XENIX. There are disadvantages to having the controller at the device driver level, however. First is software development time. Drivers must be physically linked to the XENIX kernel. This takes about 15 minutes and substantially increases the development time for the controller code. Secondly, since device drivers have full access to the system, programming errors may destroy the software system (requiring XENIX to be reloaded from diskette). In spite of these problems, this still seems to be the most practical way of building a controller in XENIX.

7. Application and Conclusion

7.1 Application

Once the Joint Interface Board was constructed and debugged, the basic I/O drivers were installed into the XENIX system and tested. After the basic system became operational, a simple but complete example of a controller was designed and tested.

The objective of this project was to design and construct the hardware and interface software to implement a robot controller. To perform a functional test on the entire system, a simple six-axis P.D. digital controller was implemented. In addition to testing the integrated system, it also served as a documented application guide for use of the JIB and the XENIX interface.

Figure 4 shows the basic controller system. The controller is divided into five distinct subsystems: (1) the application software which issues high level joint motion commands (kinematics, path planning, etc.) and runs in the normal application environment of XENIX; (2) kernel level driver software which interprets the read and write commands from the application programs; (3) interrupt level driver software which "services" the timer interrupt by executing the control structure software, reading and writing directly to the JIB hardware; (4) the Joint Interface Board which interfaces the computer joint motor signals; and (5) the robot arm itself, including power amplifiers, joint motors and feedback elements.

The simple P.D. controller implemented in this project was able to satisfactorily control all six PUMA 560 joint motors simultaneously. The P.D. coefficients were experimentally determined by trial and error. This was done one joint at a time while the other joints were locked. When all joints were operated together, the strong coupling between joints 2 and 3 (shoulder and elbow) caused strong oscillations. The gains of these joints were reduced to produce a more stable system. This is an area where more sophisticated control techniques should produce improved results.

7.2 Conclusion

The basic objective of designing and constructing a general purpose robotic controller was completed successfully. The system has been used to control the PUMA 560 robot arms, demonstrating the functionality and flexibility of the design. The Joint Interface Board has served its overall design objective well.

Using the XENIX operating system was done with mixed results. High level software is easily developed (at least for UNIX users). Whereas the method of low-level servo-loop software programming was somewhat less than desirable in that routines on this level must be directly linked (using the 'ld' linked) to the XENIX kernel. Therefore, it involves a fairly time-consuming task. XENIX also prohibits writing C-code in the kernel level which uses the floating point coprocessor (via an undocumented c-compiler flag). This was disappointing, but there are ways around this problem. This last issue is an area where more time and effort would be useful.

8. References

[1] D. G. Bihn, A Universal Six Joint Robot Controller, M.S. Thesis, Department of Electrical and Computer Engineering, University of California, Davis, 1986.

[2] Intel Corporation, System 310 Installation an Operation Guide, O.N. 173211-002, October 1983.

[3] Intel Corporation, XENIX 286 Reference Manual, O.N. 174390-008, 1984.

[4] Intel Corporation, Introduction to the iAPX 286, O.N. 210308-001, 1985.

[5] Intel Corporation, Introduction to the iAPX 287, 1985.

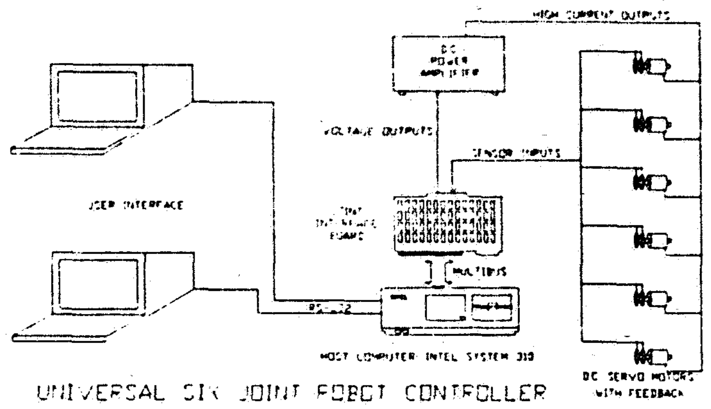
[6] Intel Corporation, Intel Multibus Specification, O.N. 9800683, 1978.

[7] Analog Devices Inc., "Fast, Complete 12-bit A/D Converter with Microprocessor Interface," Data-Acquisition Databook 1984, Volume I Integrated Circuits, 10-55, 1984.

[8] Intel Corporation, "8254 Programmable Interval Timer," Microsystem Components Handbook, Volume II, 5-240.

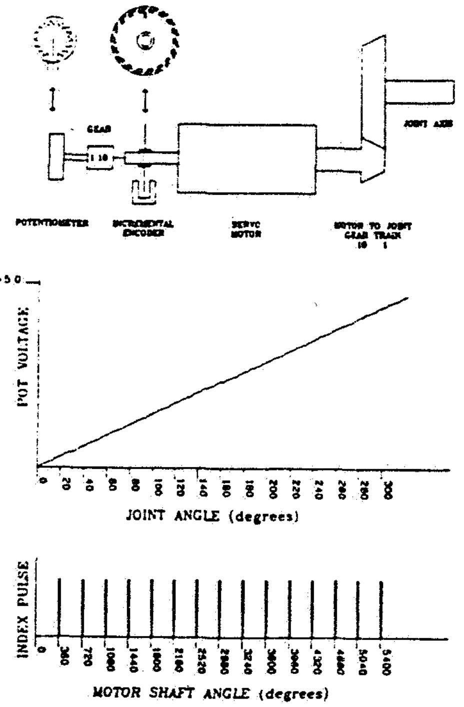
[9] Unimation, "500 Series Electrical Drawing Set for VAL II and VAL Plus Operating Systems," Unimate PUMA Mark II Robot, 394AC1, July 1985.

[10] Intel Corporation, XENIX Device Driver Guide, O.N. 174393-001, 1985.



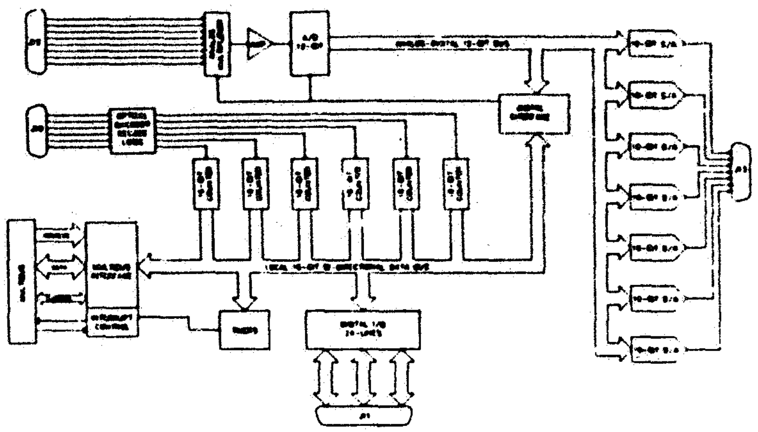
UNIVERSAL SIX JOINT ROBOT CONTROLLER

Figure 1



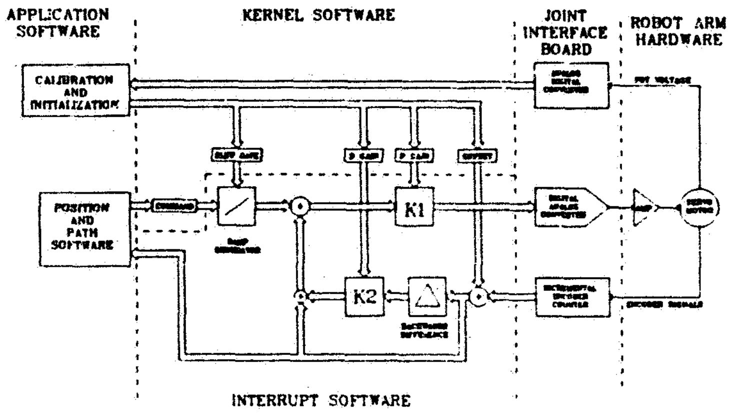
TYPICAL JOINT MOTOR CONFIGURATION

Figure 2



JOINT INTERFACE BOARD: BLOCK DIAGRAM

Figure 3



CONTROLLER LOGICAL DIAGRAM

Figure 4

S24-61
118

188144

Real-Time Graphic Simulation for Space Telerobotics Applications

E.W. Baumann
McDonnell Douglas Aerospace Information Services Co.
St. Louis, MO 63166

MP532738

~~Abstract:~~ Designing space-based telerobotic systems presents many problems unique to telerobotics and the space environment, but it also shares many common hardware and software design problems with Earth-based industrial robot applications. Such problems include manipulator design and placement, grapple-fixture design, and of course the development of effective and reliable control algorithms.

Since first being applied to industrial robotics just a few years ago, interactive graphic simulation has proven to be a powerful tool for anticipating and solving problems in the design of Earth-based robotic systems and processes. Where similar problems are encountered in the design of space-based robotic mechanisms, the same graphic simulation tools may also be of assistance.

~~This paper describes~~ the capabilities of PLACE, a commercially available interactive graphic system for the design and simulation of robotic systems and processes is described. A space-telerobotics application of the system is presented and discussed. Potential future enhancements are described.

1. Introduction

As the number and complexity of robot applications increase, the importance of being able to effectively evaluate design and programming alternatives will also increase. While such evaluation can, in most cases, be performed on a trial-and-error basis in the "real world", there are advantages to be gained by first testing those ideas in a simulated environment using computer graphics. The major advantages are:

* The time and materials spent physically prototyping alternative robotic systems is reduced or eliminated.
* The possibility of inflicting physical harm to personnel or equipment in the event of a programming error is reduced.
* Characteristics of the space environment that cannot be physically reproduced on Earth may be amenable to computer simulation.

The major disadvantage of using computer simulations to develop robotic systems and processes is that simulations are never perfect representations of what will happen in the real world. The user must therefore be careful to understand which aspects of the real world behavior are important to his application and how well they are reproduced by the simulation.

The following portions of this aper discuss some areas of robotics in which graphic simulation tools can be of value and describe several products produced by McDonnell Douglas for this purpose.

2. Conceptual Design Using Graphic Simulation

Before detailed design work can begin on a new robot application, it is first necessary to develop a general concept of the system and processes that will be required to achieve the specified goal within a particular environment. This conceptual design is useful not only as the initial step in the top-down design of a new robotic system, but also as a means to effectively describe your concepts to others. It is generally easier to get an idea across by viewing the simulated system in action than by reading pages of text and static drawings.

PLACE (Positioner Layout and Cell Evaluator) was the first in a series of McDonnell Douglas Robotics Software Products. It executes on DEC VAX 11/780, 11/750, and Micro-VAX computers using an Evans & Sutherland PS300 Computer Graphics System. The PLACE software is designed to graphically create, analyze, and modify robotic "work-cell" descriptions. A "work-cell" description is a collection of CAD-generated geometry representing the components of a robot-based manufacturing system or "work-cell". These components include robots, end-effectors, fixtures, NC machines, raw material, completed parts, and miscellaneous tooling. The designer has the option of creating an original cell description or using McDonnell Douglas supplied robots or work-cells found in the library of cell description files. These files contain models of many commercially available robots, as well as cells for a variety of robot applications.

PLACE includes the following features:

* Kinematic equations to simulate the motion of over 100 industrial robots.
* Continuous readout of joint angle data during robot motion.
* 3-D graphics for manipulation, motion specification, and visual collision detection (automatic collision detection is also available).
* Interactively controlled dynamic 3-D scaling, translating, and rotating of parts, devices, and the entire cell.
* Recording of robot motion sequences for playback and analysis.
* The ability to define attachment/detachment of parts.
* Simulation and programming of device I/O.
* Sensor support.
* Conditional execution based upon internal computation or device I/O.
* Parallel, coordinated device motion.
* A user-expandable library of robots and other work-cell components.

The addition of new robots to the library referred to above has been greatly simplified by a software package called "BUILD". BUILD automatically determines the kinematic equations of a robot manipulator from its geometric model, thereby eliminating the need to perform a custom kinematic analysis for each new robot. This makes it easy for the user to test many different manipulators or many variations of the same manipulator in order to approach an optimal design for the task to be performed. Since BUILD is limited to devices having six degrees of freedom or less, the PLACE system includes the ability to define "Compound Devices" comprised of suitably coordinated sub-devices. In this way mechanisms having greater than six coordinated degrees of freedom can be simulated.

3. Off-Line Programming

Off-line programming of robotic systems may eventually prove to be the most important application of graphic simulation tools. As robots are required to perform increasingly complex tasks in less structured environments, greater emphasis will be placed on the sensing and logical control aspects of robot programs. One way to generate such programs is to combine motion sequences produced by a system like PLACE with the remainder of a program written in the robot's native language. In this way an off-line program can be created that will already have the robot motion portions largely debugged.

McDonnell Douglas has produced a system called "COMMAND" that provides a set of translators for generating off-line programs from motion sequences created using PLACE. The translators also process instructions entered in the robot's native language. These instructions can include references to the motion sequences defined previously in PLACE. Translator output consists of a robot source program and an object code data file. This data file can be automatically written to tape or diskette as required for loading into the robot controller.

4. Space Telerobotics Example

Specific questions in the realm of space telerobotics that could be resolved, at least in part, by someone using PLACE and BUILD include:

* Can an "off-the-shelf" industrial robot be used for a space-based application? If not, then can a slightly modified version be used?
* Can a modular, reconfigurable manipulator capable of supporting a wide range of assembly and repair tasks be designed?
* Can a general-purpose gripper for space-based assembly and repair tasks be designed?
* Where should the base of a manipulator on the space station be located in order to perform a cooperative task with the shuttle manipulator?
* Where should the manipulator(s) be located on a teleoperated maneuvering vehicle (TMV)?
* How should the tool-bay of a TMV be organized? Can the tools be reached by the manipulator(s)?
* Where should the cameras be located? Where should they be looking during a particular stage of the process?
* If a manipulator needs to move a payload between two points, what paths are collision free and do not cause joint limit errors?

To illustrate and assess the capabilities of PLACE and BUILD as applied to the space-telerobotics domain, an example telerobotics scenario was developed and used as the basis for a PLACE demonstration. The basic scenario is shown in figures 1 through 7, plotted directly from the PLACE display. Here are descriptions of each figure and a few comments regarding the simulation at that point:

Figure 1 - Shuttle Arriving at the Space Station

All major movable components of the station are modelled as devices. This includes the solar panels, the large radio dish antenna, and the waste-heat radiator panels (below the radio dish). The large box-like structure located between the solar panels on the center truss is a hangar in which astronauts can perform satellite repair work. The primary goal at this point is to move the shuttle safely into a good position for transferring the laboratory module from the shuttle to the station manipulator. It is important that the position chosen not cause joint errors in either manipulator or require repositioning of the shuttle during the module transfer. Locations satisfying these constraints can be readily found by having the station and shuttle manipulators "track" the module as you use the simulator's control dials to change the shuttle vehicle's position while monitoring the manipulator joint displays.

Figure 2 - Laboratory Module Being Transferred From the Shuttle Manipulator to the Station Manipulator.

The main problem encountered here was in determining the locations for the two grapple fixtures on the module. The center grapple location worked well for the shuttle manipulator. Having the station manipulator grab the module on top and then moving the station manipulator's platform vertically to insert the module required minimal motion of the station manipulator.

Figure 3 - Teleoperated Maneuvering Vehicle (TMV) Preparing to Capture a Satellite.

Each of the TMV's main arms has six degrees of freedom. Each finger has five degrees of freedom. The arms and fingers can be controlled independently or as a single "Coordinated Motion Compound Device". The vehicle has two cameras, one located on the left boom and one on the right. Two disk shaped communication antennas are located behind and slightly below the camera booms. The fingers on the TMV's right hand are brought to a point for insertion into the recessed nozzle of the satellite. The fingers of the left hand are opened to form a flat surface to push against the opposite end of the satellite.

Figure 4 - TMV Transferring Satellite to Space Station Hangar.

Here the TMV is getting into position to deposit the satellite into the hangar. It's approach from "below" (between the hangar and the radiators) requires that the station's antenna be moved out of the way to avoid a possible collision. It may have been better to enter the hangar from above (where the two trusses meet) but in that case there might still be a need to reposition the solar panels to reduce the chance of collision.

Figure 5 - Station Climbing, Observing, and RePair (SCORP) Vehicle Attached to Station Manipulator.

By replacing the three-fingered hands with cylindrical grippers suitable for grasping space station struts, replacing the fixed cameras on booms with movable cameras on "eyestalks", and by adding a "tail" capable of securely latching onto any portion of the station's truss structure, the TMV can be converted into a vehicle capable of climbing on, inspecting, and repairing the space station. It is shown here attached to the station manipulator prior to being placed on the truss structure. The SCORP has no engines and therefore must always be attached to the station in some way. A parts bay is located inside the SCORP's "chest" below its left arm. A tool bay is located below its right arm.

Figure 6 - SCORP Climbing on Station While Performing Visual Inspection

Climbing is accomplished by declaring the arms and body of the vehicle to be a "Coordinated Motion Device", thereby forcing them to begin and end their motion simultaneously. The requirement that one hand continue to grasp the station while the body and arms are being repositioned is indicated to the PLACE system by temporarily declaring the corresponding arm to be a "Dependent Motion Device". The system then asks the user to specify the spatial relationship (in this case between hand and strut) that must be maintained during the execution of this climbing step. Once all of the goal positions and dependencies are defined, the simulation can proceed. The same approach can be used to simulate walking.

Figure 7 - SCORP Anchored to Station While Welding Reinforcement Strut to Space Station Box-Truss Structure.

When a repair job requires the use of both arms, the SCORP's "tail" can be used to grab the station structure, thereby freeing the arms while still providing substantial local mobility for the vehicle. The grapple position shown in this figure actually causes the tail-arm to exceed two joint limits in its wrist and should be changed. Exceeding joint limits in this way is a common error that would be difficult and time consuming to identify without the use of a simulation system like PLACE.

5. Future Directions

The primary emphasis of these "first generation" graphic simulation systems has been on providing user-friendly methods for specifying robot motion, and then accurately portraying the programmed motion. There is no doubt that additional improvements can, and will, be made in these areas.

As more complex robot applications appear, greater attention will be paid to simulating sensor-based robot behavior and in keeping track of accumulated position and force errors that may lead to failure. Automatic generation of "sensory expectations" for the robot's sensory systems will also be necessary if a complete off-line programming environment is to be realized.

So far these systems have only acted as providers of relatively raw information to a human decision maker. In the not-too-distant future we may see systems having the ability to analyze their own simulation results and give advice to the human user during a cell design or programming session. Perhaps planning some parts of the process, such as finding collision-free manipulator trajectories, will gradually be turned over to the system, with the human user acting increasingly as "supervisor" rather than programer. Ultimately there will cease to be a need for graphic output from the robot simulation and planning system, except to serve as a window into the machine's planning process as it determines how to accomplish to goals we have set before it.

6. Acknowledgments

I would like to thank Nicholas Denissen for his assistance in creating several of the Unigraphics CAD models used in the space telerobotics example.

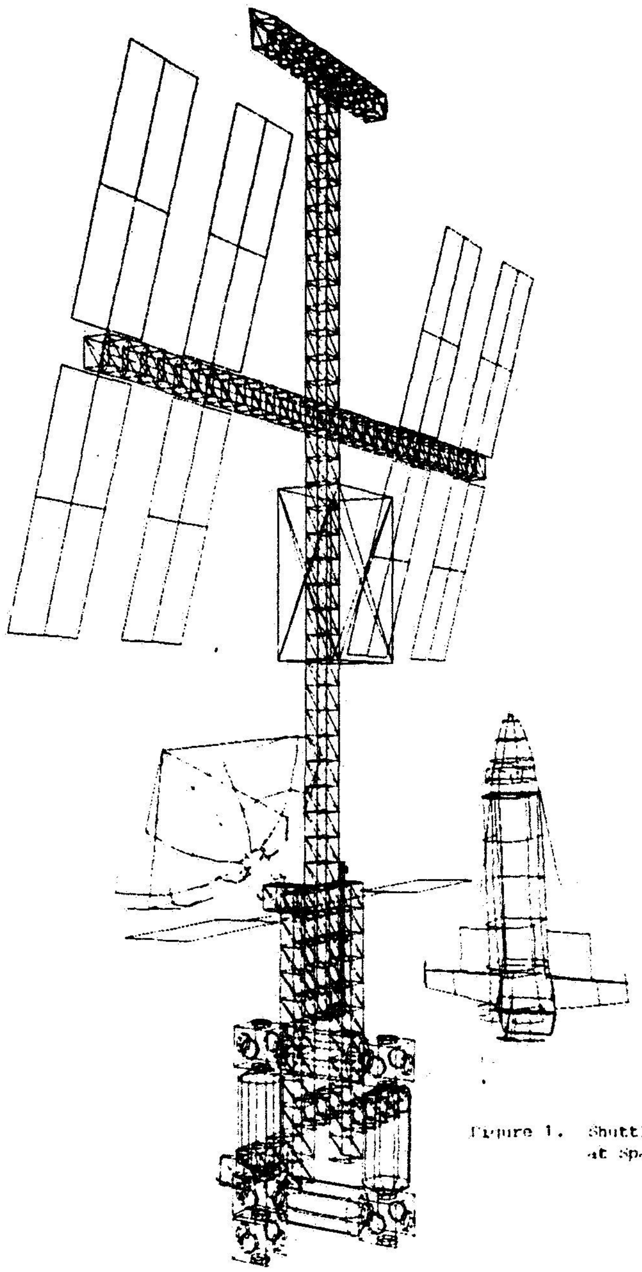


Figure 1. Shuttle Arriving at Space Station

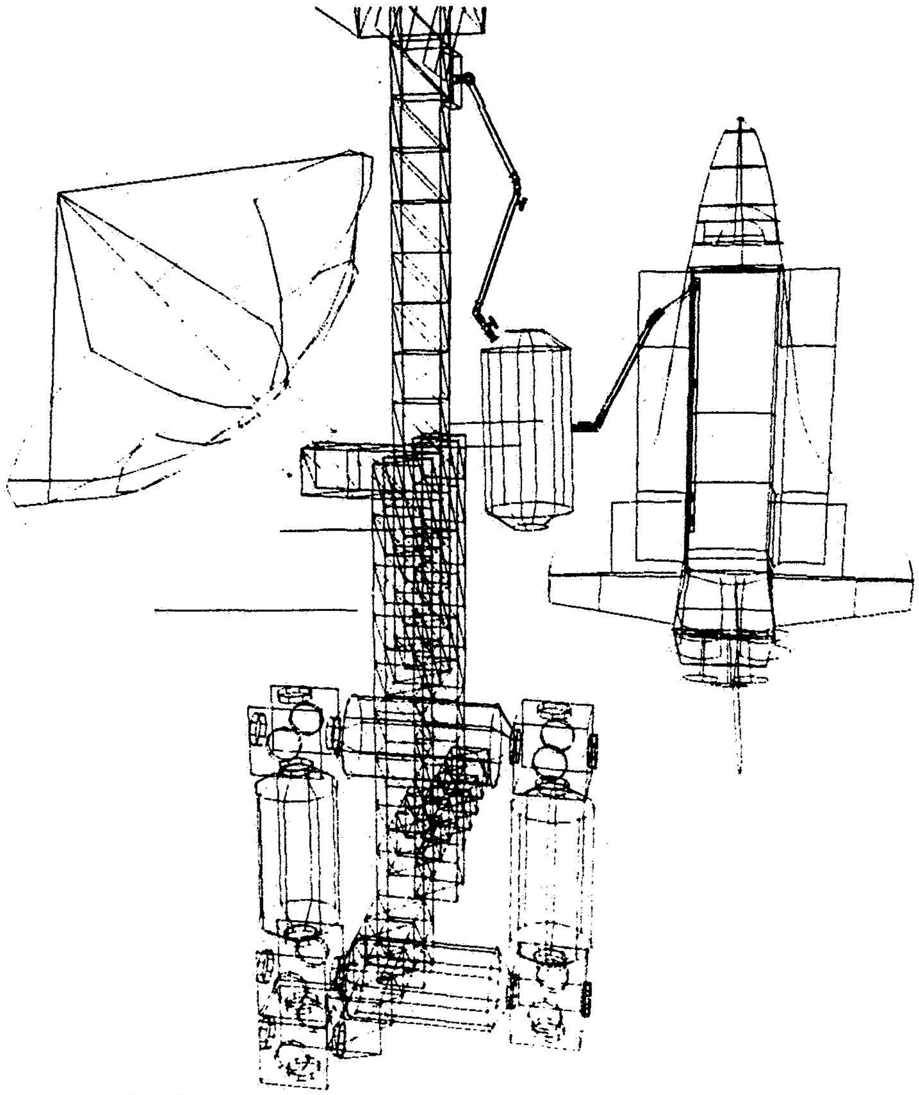


Figure 2. Laboratory Module Being Transferred From Shuttle Manipulator to Station Manipulator

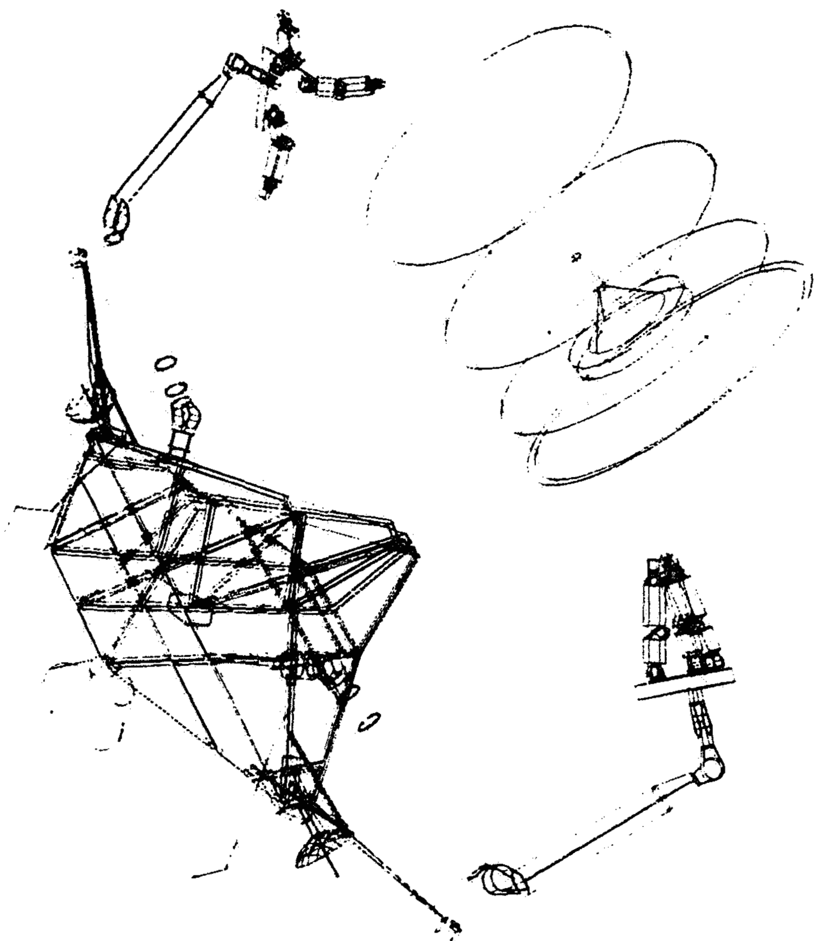


Figure 3. Teleoperated Maneuvering Vehicle (TMV) Preparing to Capture a Satellite

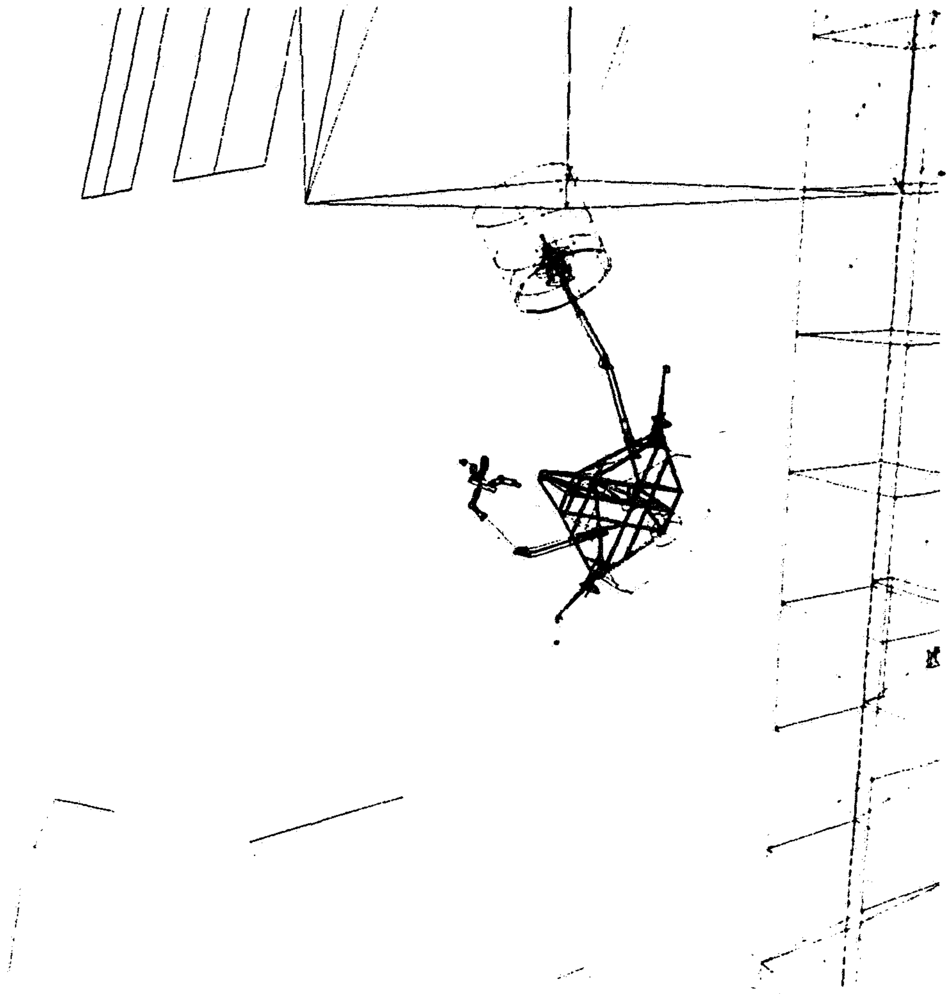


Figure 4. TMV Transferring Satellite to Space Station Hangar

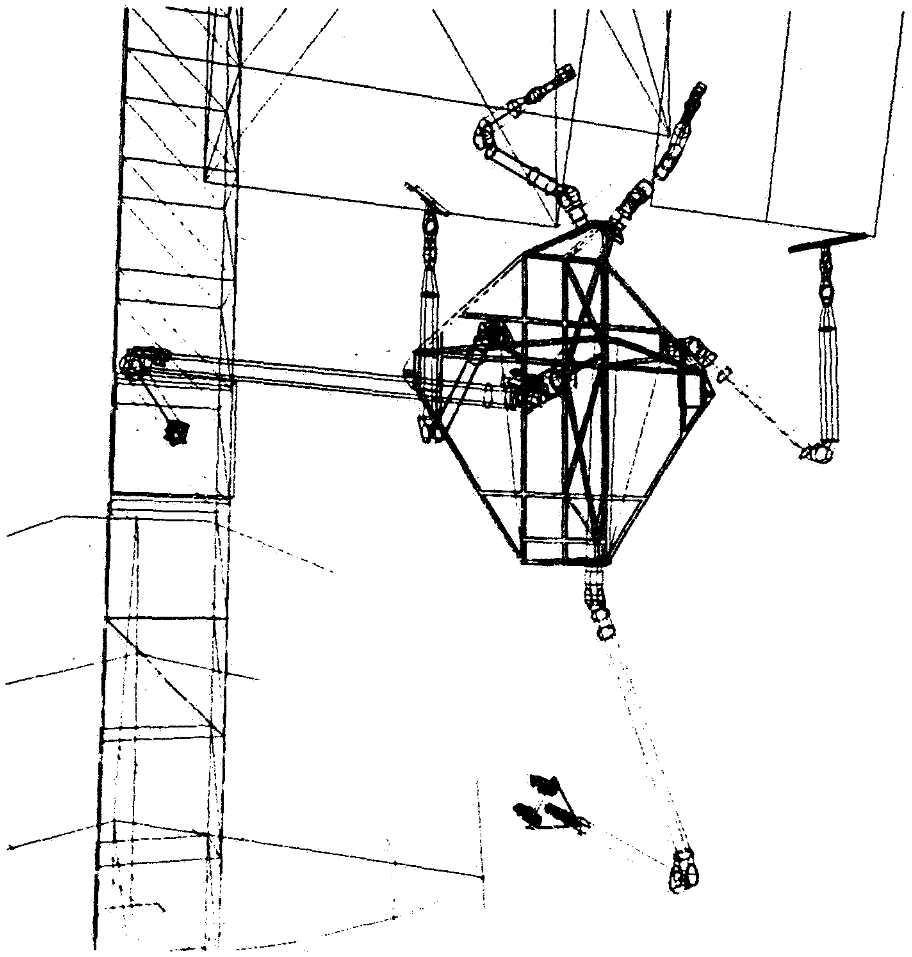


Figure 5. Station C[illegible] [illegible]ing, Observing, and RePair [illegible] Vehic[illegible] (SCORP) Attached to [illegible]ation Manipulator

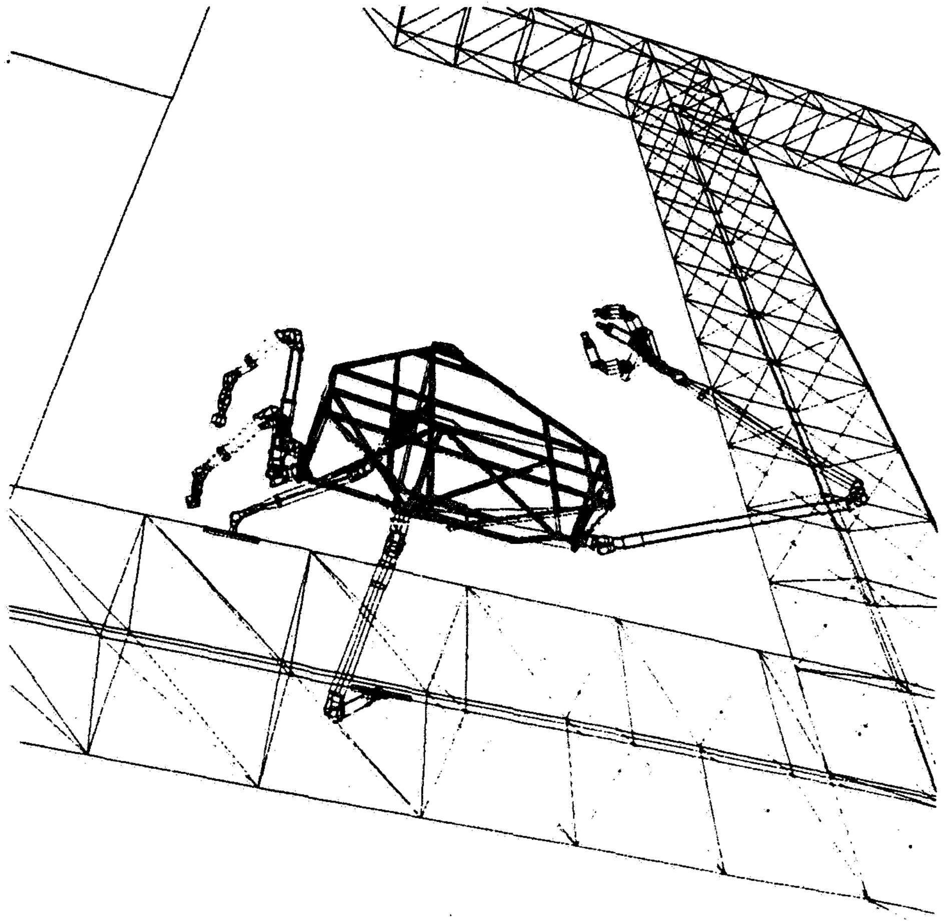


Figure 6.SCORP Climbing on Station While Performing Visual Inspection

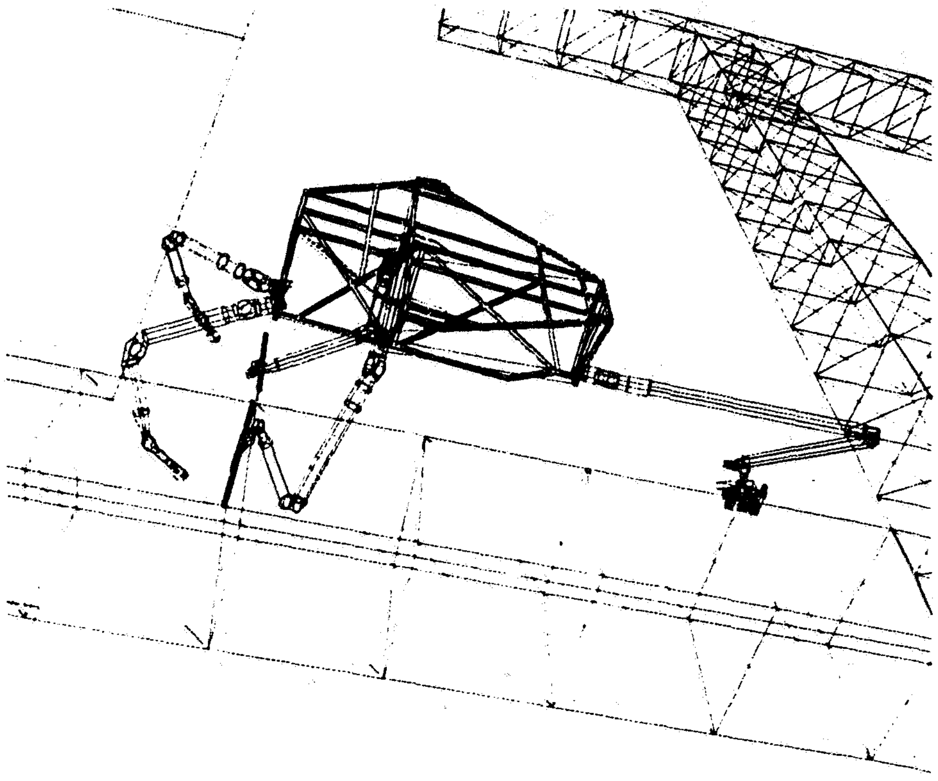


Figure 7. SCORP Anchored to Station While Welding Reinforcement Strut to Space Station Box-Truss Structure

S25-37
9P.

188145

Manipulator Control and Mechanization: A Telerobot Subsystem

S. Hayati and B. Wilcox
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109

JJ574450

1. ABSTRACT

~~This paper describes~~ the short- and long-term autonomous robot control activities in the Robotics and Teleoperators Research Group at the Jet Propulsion Laboratory (JPL) are described. This group is one of several involved in robotics which is an integral part of a new NASA robotics initiative called Tele obot program. ~~This paper provides~~ a description of the architecture, hardware and software, and the research direction in manipulator control is given.

2. INTRODUCTION

The Telerobot program is a new project initiated in 1985 by NASA. The aim of this program is to develop a technology base in the areas of teleoperators, robotics, human factors, artificial intelligence, vision and other sensors, and manipulators. The objective is to develop and integrate the technologies to be used in future NASA endeavors, particularly for on-orbit assembly, maintenance, repair, and operation. To realize the goals of the program, JPL and other NASA centers have been funded to develop core technologies with broad applications in automation and robotics and to carry out a series of ground demonstrations of the developed technologies. These demonstrations are currently planned for 1988, 1990, 1993, and beyond. Each successive demonstration will evidence proof-of-concept for a higher degree of autonomy tnan its predecessor. The short-term objectives are set forth by the first demonstrator in 1988. This paper will give a detailed description of the hardware, software, and control strategies that have been planned to carry out the 1988 demonstration task. The long-term goals of the group's activities will also be described.

3. TELEROBOT ARCHITECTURE

A testbed is required as a general facility to test and validate theoretical developments at JPL and other NASA centers. JPL has developed a flexible and hierarchical system architecture for the Telerobot Testbed facility. Figure 3.1 illustrates the major components of this architecture. It is recognized that in the foreseeable future human intelligence will be required for complex robot task execution. The architecture is designed so that the operator can assume control or halt the autonomous task execution at any time. Certain provisions were necessary to eliminate the risk of damaging the workpieces or the manipulators by prohibiting the operator from halting the autonomous operation in some critical instances. For example, stopping the autonomous activity during a satellite capturing task could possibly damage either the arms or the satellite or both. In this particular instance the autonomous operation will acknowledge the operator's desire to stop the operation but will first execute a routine to withdraw the arms to a safe position before bringing them to a complete stop. An overview of this architecture is documented in reference [1].

On the autonomous side, the AIP (Artificial Intelligence Planner) will develop task scripts from requests made by the operator and will specify certain regions of space in which the arms must be moved based on global spatial planning. In the near-term, most of the AIP activities will be off-line. It is envisioned that the AIP will have on-line task planning and error recovery in the future.

Run Time Control (RTC) is the second subsystem in the hierarchy. This subsystem serves several important functions in the autonomous operation mode. It will receive high level task planning information from the AIP and break them down to a number of primitive operations that can be executed in the Manipulator Control and Mechanization (MCM) subsystem. This subsystem will determine collision free paths for the robot and select an appropriate one to avoid wrist and workspace singularities. RTC will keep track of the world model and update it as the manipulators modify the geometry of the environment. This subsystem will coordinate other subsystems to realize a particular task. A more detailed description of this subsystem is given in references [2] and [3].

Sensing and Perception is a subsystem which will provide acquisition and tracking capability for the tracking of known but unlabelled moving objects and position verification for fixtures on workpieces (e.g. bolts, handles, etc.). The vision system currently under development includes custom-designed image-processing hardware, and acquisition and tracking software running on a general purpose computer. More detailed information on this subsystem and its activities are documented in references [4] and [5].

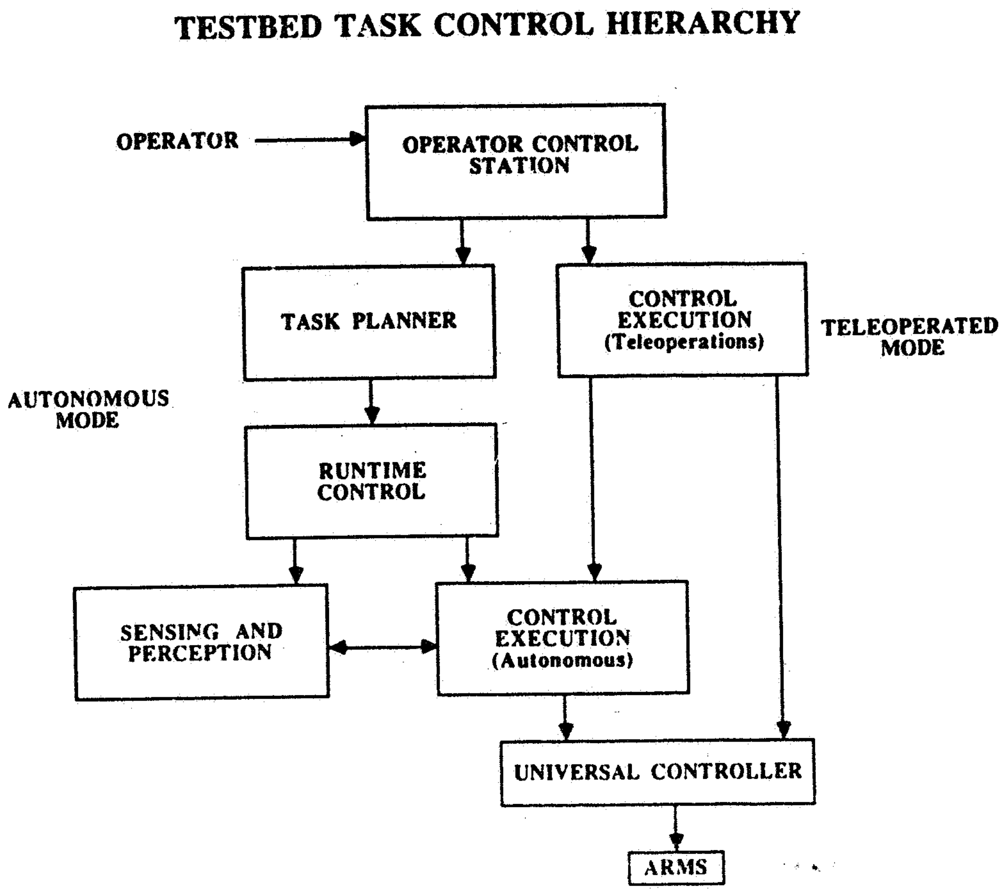


Figure 3.1 Testbed Task Control Hierarchy

Manipulator control and Mechanization (MCM) is the subsystem that is responsible for trajectory generation and low-level control of the manipulators in the autonomous mode of operation. Sections 4 and 5 will provide a detailed description of this subsystem and current research activities.

The teleoperator subsystem forms a parallel link to the autonomous hierarchy so that the operator can control the manipulators directly. The control is based on the operator generating commands by physically moving two six degree-of-freedom (DOF) force reflecting hand controllers with the remote site manipulators responding to these commands. The hand controllers themselves are six DOF manipulators with DC motors to realize force reflection, and use a distributed microprocessor computing architecture. References [7] through [9] provide a more detailed description of this subsystem.

4. MANIPULATOR CONTROL AND MECHANIZATION SUBSYSTEM

The goal of this subsystem is two-fold. It is designed to 1) provide low-level robot control for the Telerobot testbed facility and 2) furnish a research facility for testing robot control algorithms. The selection and design of the software and hardware for this subsystem were based on several factors, among which portability and extendibility were critical. Although when viewed from the Telerobot system level, MCM can be considered to be a low-level system, MCM itself has several levels of hierarchy. The software is based on a robot language, RCCL (Robot Control "C" Library), developed at Purdue University by Professors Richard Paul and Vincent Hayward [10]-[12]. A brief description of the software architecture is given later in this section.

The manipulator hardware at the present time consists of three PUMA 560 robots. One of the arms will serve as a platform for positioning and orienting a pair of stereo cameras for the Sensing and Perception subsystem. The other two arms, which will be used for single and dual arm manipulations, are mounted on lathe beds so their relative distance can be modified to accommodated various task requirements. In the future this system will be mechanized to provide servo controlled simultaneous relative positioning of the manipulators' single and dual arm operations. This will increase the work volume of the manipulators and will bring about challenging theoretical problems both in task planning and cooperating arm control. The manipulation arms are equipped with commercial (LORD Corporation) force-torque sensors with associated microprocessors. These arms are also currently equipped with simple on-off pneumatic grippers.

The testbed includes a 350 pound satellite mockup which can spin and nutate freely on a gimbal for up to several minutes, closely simulating the dynamics of a real satellite. The satellite mockup is fitted with a panel which is affixed to one of its sides by means of four screws. The removal of the panel can best be accomplished by two cooperating arms after the screws are removed. The task complexity can be increased by mounting various elements under this panel, such as PC boards and electrical connectors with cables attached. The satellite mockup is also fitted with an (EVA) fluid connector, which is a coupling device designed for transferring fluids and low pressure gases. The assembly and removal of this coupler also introduces single and dual arm force/position control problems that must be dealt with. The setup presents many realistic and complex problems for robot task planning and control. One challenging task is to track the position/orientation of the slowly spinning satellite by the Sensing and Perception subsystem, grapple with the satellite and bring it to a rest position without exerting excessive forces/torques on the arms. This task requires cooperative arm control as soon as the arms come in contact with the satellite. Figure 4.1 shows the MCM testbed facility.

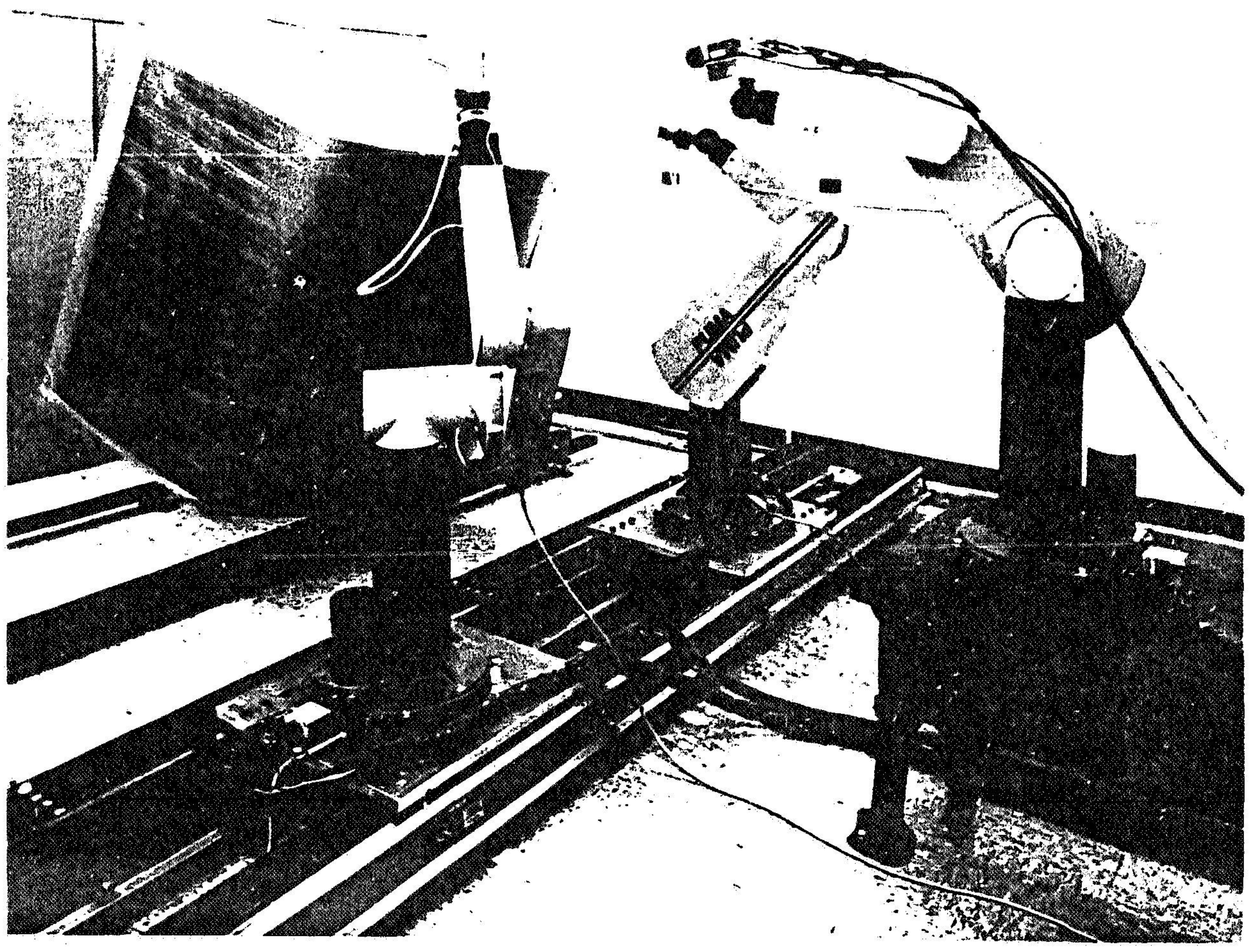


Figure 4.1 Testbed Facility at the Robotics and Teleoperators Research Group

The computing facilities at the present time are a MicroVax II, three Unimate controllers and the microprocessors of the force torque sensors. Figure 4.2 illustrates the detailed hardware schematics of MCM and its interface with RTC and Sensing and Perception. Since RCCL plays a central role in the MCM subsystem, a brief description of the language and its capabilities and limitations will be discussed below. For more detailed information see references [13] and [14].

The system software consists of a series of programs running simultaneously on various processors. Figure 4.3 shows a block diagram of the RCCL architecture. The configuration uses the Unimate controllers as low-level servo control units. The LSI 11/73 microprocessor in the Unimate controller is utilized as an I/O system to link the MicroVax II to the 6503 joint microprocessors. A hard clock constantly interrupts the I/O control program at a preselected sample rate. At every interrupt, a program which resides in the LSI 11/73 gathers information about the state of the robot arm, including joint positions and currents, front panel switch register contents, A/D converter readings, parallel port data, and teach pendant signals. The program then

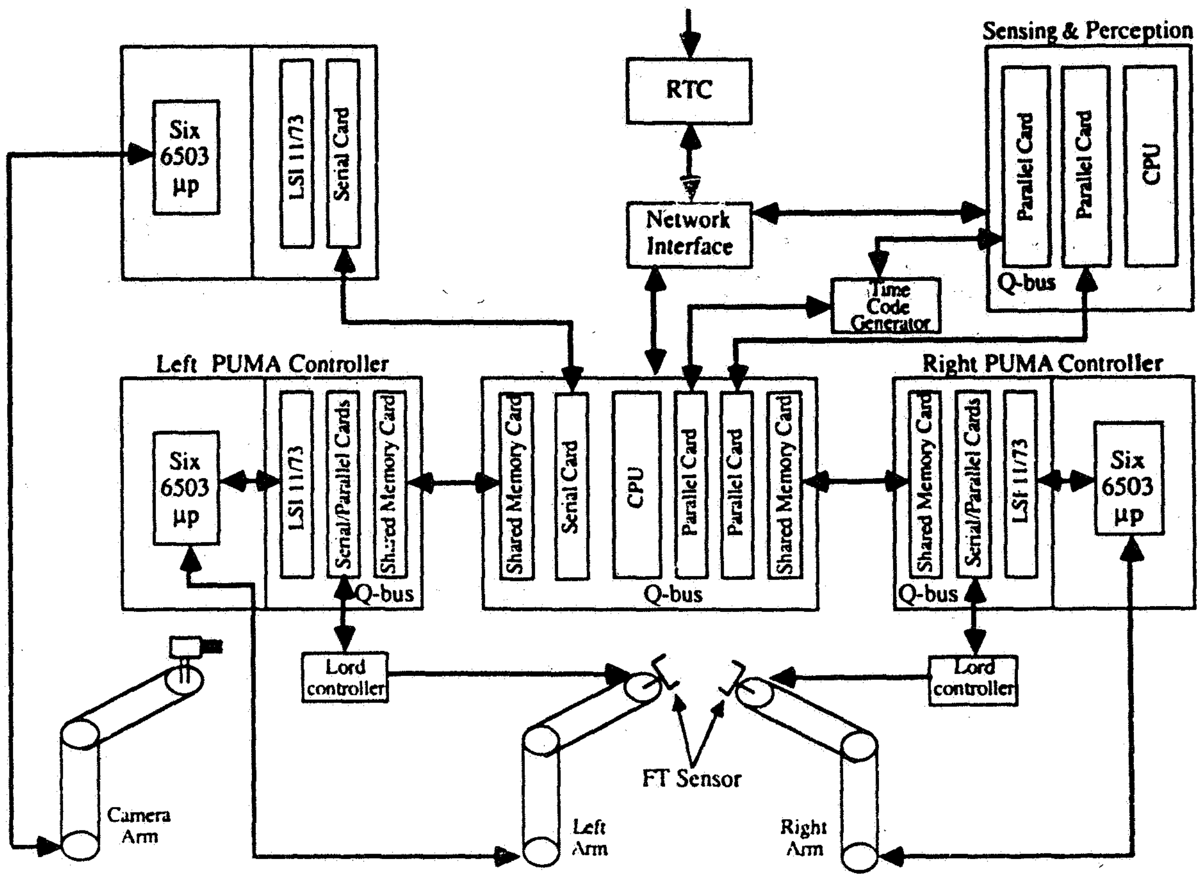


Figure 4.2 Detailed MCM Hardware and Interface

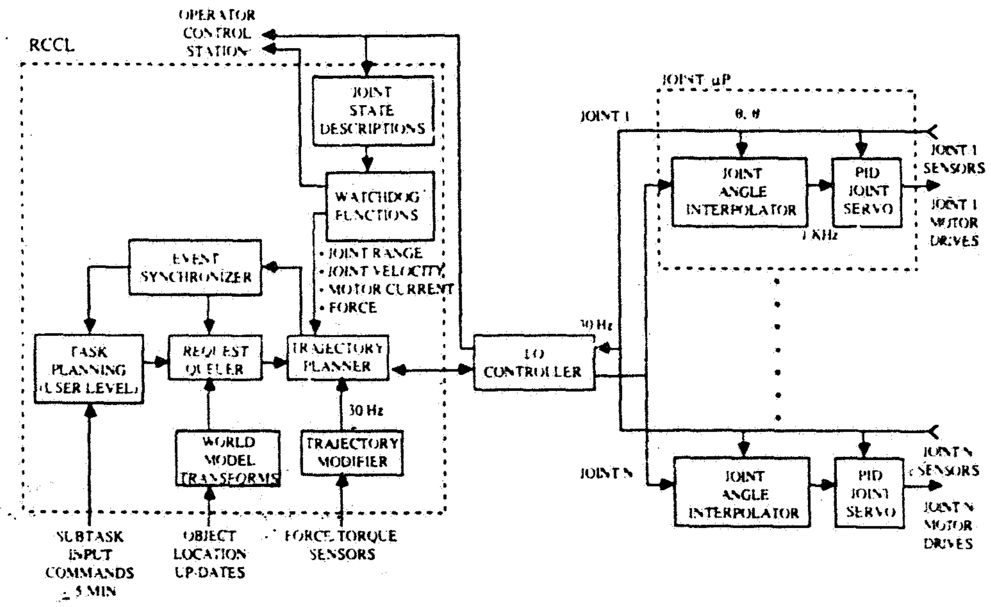


Figure 4.3 Functional Diagram of RCCL and Unimate Controller Resident Software

interrupts the control level on the MicroVax II, transmits this data, waits for the control level to return a set of joint commands, and then dispatches these commands to the required joints. The sample rate can be changed from its normal setting of 28 msec to 56, 14 and 7 msec.

The MicroVax II contains the planning and control programs, which run concurrently with each other. The planning level, which interacts with the user, operates in the normal time-sharing context and has access to all standard resources, such as files, devices, and system calls. The user, utilizing the library functions, specifies by a Cartesian frame the goal position and via points that the end-effector must pass through. The planning level forms a motion queue based on the sequence in which the user has specified the motions. High-level functions are available to change the sample rate and modify the planned path in real-time based on either an internally generated path modifier or by use of external sensors.

The control level runs in the foreground and executes a number of procedures at the sample rate of the system. When it is interrupted by the LSI 11/73 it first checks the received information for data integrity and the normal status of the arms' joint servos. The data consists of joint angle readings, motor currents, and the robot's status. In the JPL implementation, the data also includes the force/torque readings received by the LSI 11/73 once every sample time. The program then transmits the new set points that this level has computed in the last sample interval through the LSI 11/73 to the joint microprocessors. It then executes a control function (see Fig. 4.4) to calculate a new set of joint servo settings. This control function is normally a trajectory generator but, as was mentioned earlier, it can also include a user function for real-time modification of the trajectory which the user has defined at the user level. To meet the constraints imposed by the sample rate, the control level executes in the highest priority mode. The set points normally are new joint positions but can also represent motor currents for force servoing.

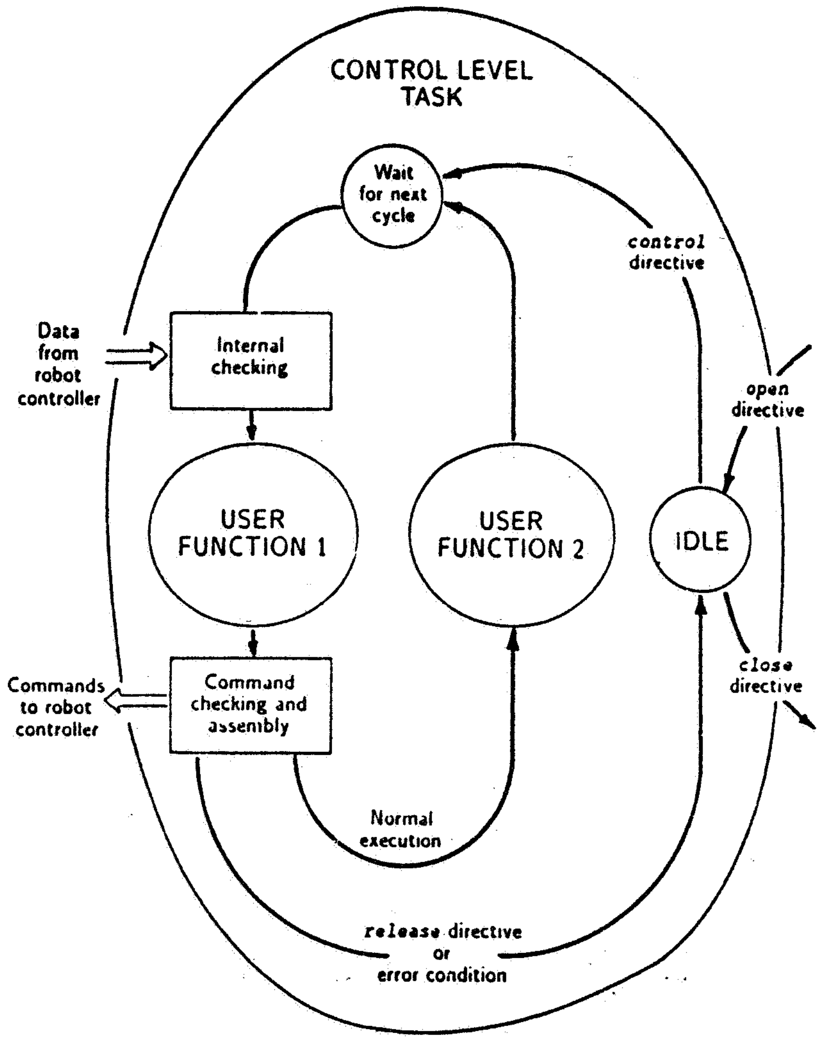


Figure 4.4 Control Level Software Block Diagram

The Telerobot subsystems will be connected to form a network with an Ethernet cable. The subsystems will communicate with each other using the 10 megabit Ethernet "physical link". Because most of the computers will be VAX's using the VMS operating system, the DECNET protocol has been selected as the basic "logical link" over the Ethernet. Since the Unix operating system does not support DECNET, an intermediate MicroVax II running under the VMS operating system is utilized as a link between the Unix MicroVax II and the other subsystems. These two microVax II's are connected to each other via a shared memory card.

Although the current setup provides a flexible and portable programming environment, there are severe problems and shortcomings that must be addressed. The current RCCL implementation at JPL is viewed as a short-term solution for the MCM subsystem. One problem with the current setup is that most sophisticated robot control algorithms require very high throughput. Presently only the kinematics of the robot is considered in generating the set points. The computation burden is on a single MicroVax II CPU which cannot meet the high throughput requirements of advanced multivariable control laws. A second problem is posed by the language, which is written for the control of a single robot arm. Any modification to the language must include the capability to plan for and control two or more arms simultaneously. A third problem lies with the Unimate controller. Although it is possible to use this controller to run arms other than Unimation's, one is limited by the speed and particular control method used in the servo controllers. In the following the we describe our plans for addressing these limitations.

Currently JPL is in the process of designing a low-level robot controller based on distributed microprocessors. Initially this controller will have the capability of controlling eight joint motors[15]. This capability can easily be extended to control more that eight joints. The first goal is to control both the PUMA 560 arms and the Universal Force Reflecting Hand Controllers. In 1989 this controller will be used to control the seven DOF space-like arms currently under development at the Oak Ridge National Laboratory under contract to the NASA Langley Research Center [16]. In addition, a distributed microprocessor-based computing facility is being developed to replace the MicroVax II computer as the MCM computer. At the present time only a preliminary design is established for this hardware. Figure 4.5 shows a preliminary block diagram of this computing facility and its integration with the joint controller system. To summarize, for 1988-1989 JPL will have three main elements for advanced manipulator control. These are 1) programmable joint controllers that can be used to control various robots, 2) an open architecture distributed microprocessor computing facility for trajectory planning and control of multiple cooperating manipulators, and 3) seven DOF modular space-like manipulators.

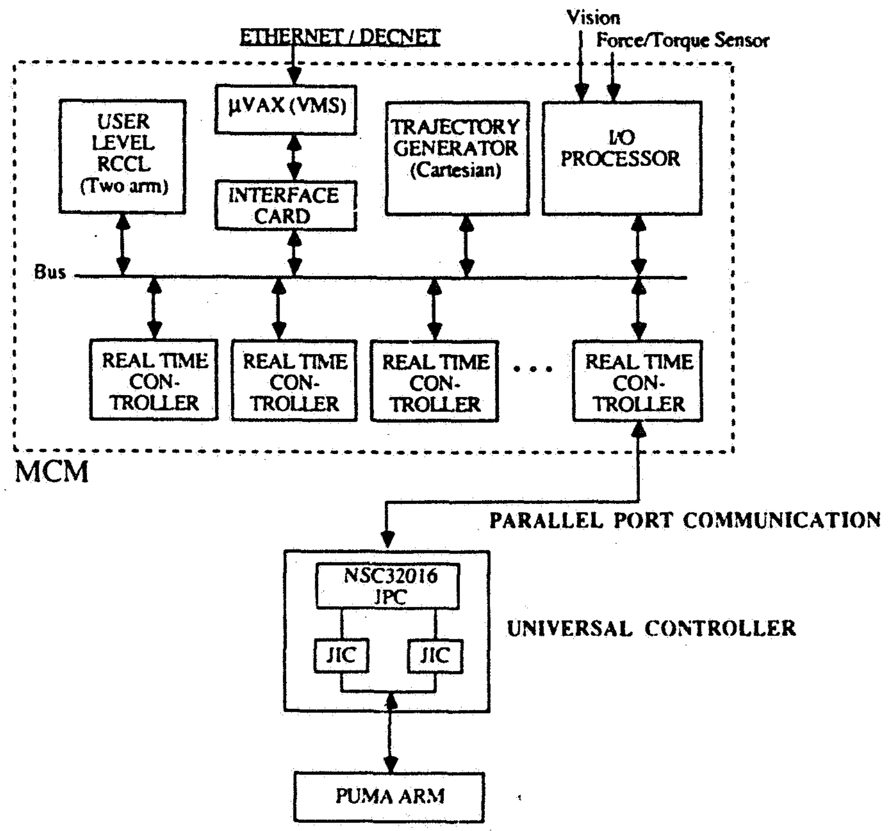


Figure 4.5 Preliminary Architecture for MCM Distributed Microprocessor Computing Facility

5. RESEARCH IN MANIPULATOR KINEMATICS, DYNAMICS AND CONTROL

Our research activity is in support of both near- and long-term goals established by the the Telerobot program. In the following we will describe the main research activities pursued by the group in manipulator control and mechanization.

5.1 Manipulator Geometry Modelling

One of the most important functions of autonomous robots is movement of their end-effectors to various locations in the work space. Tasks performed by these robots require a certain positioning accuracy. Experience with industrial robots has shown that although the relative positioning accuracy (or repeatability) is satisfactory, the absolute positioning accuracy is not acceptable. This inaccuracy is largely due to uncertainty in the manipulator's geometric parameters. Our research has resulted in a parameter identification technique to update the geometric errors of the manipulators. Both simulation and actual laboratory experiments have shown the validity of the technique.

An associated problem with the geometry calibration is the inverse kinematics problem of so called near-simple manipulators. To utilize the results obtained from the above geometric calibration one must incorporate the improved knowledge of the link parameter errors in the forward and inverse kinematics equations of the calibrated robot. Modification of the forward kinematic equations is very simple. Modification of the inverse kinematics, unfortunately, is not so easy. It is well known [17] that for a large class of robots the inverse kinematic solution can be obtained in a closed form. The condition for the existence of an analytic solution is that at least three consecutive joint axes must intersect at one point (a "simple" arm). The post-calibrated model of the robot, which more accurately represents the physical system, is that of a non-simple one. The inverse kinematic equations are solved by first finding the closed form solution for the ideal model and then computing small variations to be added to the joint angles by utilizing the Jacobian of the post-calibrated model. For more detail see references [18]-[20].

5.2 Model-based Dual Arm Control

The topic of multiple robot control is relatively new in robotics research. The extension of robot control techniques to the case of multiple manipulators is necessitated by realities encountered both for manipulating small objects and for handling large workpieces. The manipulation of objects normally requires at least two hands to simultaneously position and reorient the object so that either one or both hands can perform their respective tasks.

Our research in this area has been based on the derivation of the equations of motion in the so-called Operational Space (or Cartesian state space). We assume a general case of n cooperating robots which are holding an object rigidly. This object may also be constrained from motion in one or more dimensions by an external environment. Equations of motion are derived using the Lagrange multiplier technique. It is assumed that each manipulator is equipped with a force/torque sensor capable of measuring three orthogonal forces and torques in a given coordinate frame. The aim is to control the position of the object and its interaction forces with the environment in the sense of hybrid control of Raibert and Craig [21]. Utilizing these dynamics equations a decoupling controller in configuration space is designed to control both the position and the interaction forces of the object with the environment. Preliminary simulation studies on a simple system which consists of a pair of two-link manipulators holding a load which interacts with an environment have shown that the control technique yields excellent results. For more details please refer to references [22] and [23].

5.3 Adaptive Control of Manipulators

Adaptive control offers an appealing solution to the control problem. In adaptive robot control methods, neither the complex mathematical model of the robot dynamics nor any knowledge of the robot parameters or the payload are required to generate the control action. Adaptive control methods fall into two distinct categories, indirect and direct. In direct adaptive control methods the control action is generated directly, without prior parameter estimation. Research in this area was started by the application of adaptive control techniques to control the manipulator in joint space. Research was then extended to the control of manipulators in Cartesian space. Further research resulted in an adaptive control technique for simultaneous position and force control of manipulators. Most recently, an adaptive controller was formulated for the control of multiple cooperating robots. Simulation studies on two link manipulators have shown excellent results for all of the above adaptive controllers. Additional detail is contained in references [24]-[27].

6. CONCLUSIONS AND FUTURE RESEARCH DIRECTION

Most of the Robotics and Teleoperators Research Group's research activity in the manipulator control area is of a theoretical nature. Much effort and further research will be required to implement the proposed control algorithms. Several important realistic problems such as arm friction and backlash, joint flexibility, computational complexity resulting in low sampling rates, finite measurement resolution and measurement noise will have to be considered before a robust controller can be realized. Further theoretical work in multiple cooperative arm control and redundant arm control is currently being carried out.

7. REFERENCES

[1] P. S. Schenker, et al., "The NASA Telerobot Technology Demonstrator," Proceedings of the SPIE Conference on Advances in Intelligent Robotics Systems, Cambridge, Mass., Oct. 26-31, 1986.

[2] J. Balaram, et al, "Run-Time Research for Autonomous Robots," Proceedings of the international Symposium on Robot Manipulators: Modeling, Control, and Education, Albuquerque, New Mexico, November 12-14, 1986.

[3] A. Loshkin, et al., " A Run-Time Interpreter for a Telerobot System," Proceedings of the international Symposium on Robot Manipulators: Modeling, Control, and Education, Albuquerque, New Mexico, November 12-14, 1986.

[4] D. B. Gennery and B. Wilcox, " A Pipelined Processor for Low-Level Vision," Proc. IEEE Computer Society Conference on Computer Vision and Pattern recognition, San Francisco, Ca., June 1985.

[5] D. B. Gennery, "Stereo Vision for the Acquisition and Tracking of Moving Three-Dimentional Objects," in Techniques for 3-D Machine Perception (A. Rosenfeld, ed.), North-Holland, 1986.

[6] B. Wilcox, B. Bon, and D. B. Gennery, "Machine Vision Research at Jet Propulsion Laboratory," AIAA/NASA Symposium on Automation, Robotics and Advanced Computing for the National Space Program, September 4-6, 1985, Washington, D.C.

[7] A. K. Bejczy and J. K. Salisbury, "Kinesthetic Coupling Between Operator and Remote Manipulator," Proceedings of ASME Computer Technology Conference, Volume 1, San Francisco, CA, August 12-15, 1980; and "Controlling Remote Manipulators Through Kinesthetic Coupling, Computers in Mechanical Engineering, Vol. 1, No. 1, July 1983.

[8] S. K. Lee, G. Bekey, and A.K. Bejczy, "Computer Control of Space-Borne Teleoperators with Sensory Feedback," 1985 IEEE International conference on Robotics and Automation, St. Louis, MO, March 25-28, 1985.

[9] C. P. Fong, R.S. Dotson, and A.K. Bejczy, "Distributed Microprocessor Control System for Advanced Teleoperation," 1986 IEEE International Conference on Robotics and Automation, San Francisco, Ca., April 7-10.

[10] V. Hayward, "RCCL User's Manual, Version 1.0," Purdue University Technical Report, TR-EE 83-46.

[11] V. Hayward and R. Paul, "Introduction to RCCL: A Robot Control C Library," First IEEE International Conference on Robotics, Atlanta, Georgia, 1984.

[12] V. Hayward and R. Paul, "Robot Manipulator Control Under Unix:RCCL a Robot Control C Library," International Journal of Robotic Research, Vol. 5, No. 3, 1986.

[13] J. Lloyd, "Implementation of a Robot Control Development Environment," Master's thesis, Electrical Engineering Department, McGill University, Montreal, Canada, 1985.

[14] J. Lee, S. Hayati, et al., "Implementation of RCCL, a Robot Control C Library on a MicroVax II," Proceedings of the SPIE Conference on Advances in Intelligent Robotics Systems, Cambridge, Mass., Oct. 26-31, 1986.

[15] A. K. Bejczy and S. F. Zoltan, "A Universal Computer Control System (UCCS) for Space Telerobots," Proceedings of the IEEE International Conference on Robotics and Automation, Raleigh, North Carolina, March 30 to April 3, 1987.

[16] H. L. Martin, et. al., "Recommendation for the Next-Generation Space Telerobot System," a publication by Oak Ridge National Laboratory, ORNL/TM-9951, 1986.

[17] D. Pieper, "The kinematics of Manipulators Under Computer control," Ph.D. Thesis, Stanford University, 1968.

[18] S. Hayati, "Robot Arm Geometric Link Parameter Estimation," Proceedings of the 22nd IEEE Conference on Decision and Control, San Antonio, Texas, December 14-16, 1983.

[19] S. Hayati and M. Mirmirani, "Improving the Absolute Positioning Accuracy of Robot Manipulators," Journal of Robotic Systems, Vol. II, No. IV, 1985.

[20] S. Hayati and G. Roston, "Inverse Kinematic Solution for Near-Simple Robots and its Application to Robot Calibration," Proceedings of the International Symposium on Robot Manipulators: Modeling, Control, and Education, Albuquerque, New Mexico, November 12-14, 1986.

[21] M. Raibert and J. Craig, "Hybrid Position/Force Control of Manipulators," ASME Journal of Dynamic Systems, Measurement, and Control, June 1981.

[22] S. Hayati, " Hybrid Position/Force Control of Multi-Arm Cooperating Robots," Proceedings of the 1986 IEEE International Conference on Robotics and Automation, San Francisco, Ca., April 7-10, 1986.

[23] S. Hayati, "Dynamics and Control of Coordinated Multiple Manipulators," Proceedings of the Workshop on Space Telerobotics, Ca., 1987.

[24] H. Seraji, "Design of Adaptive Joint Controllers for Robots," Proceedings of the International Symposium on Robot Manipulators: Modeling, Control, and Education, Albuquerque, New Mexico, November 12-14, 1986.

[25] H. Seraji, "Direct Adaptive Control of Manipulators in Cartesian Space," Journal of Robotic Systems, Vol. 4, No.1, Feb., 1987.

[26] H. Seraji, "Adaptive Force and Position Control of Manipulators," Journal of Robotic Systems, August 1987 (to appear).

[27] H. Seraji, "Adaptive Control Strategies for Cooperative Dual-Arm Manipulators," Proceedings of the Workshop on Space Telerobotics, JPL Publication 87-13, Jet Propulsion Laboratory, Pasadena, Ca., 1987.

8. ACKNOWLEDGMENT

The research described in this paper was performed at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

S26-37
68
188146

Chaos Motion in Robot Manipulators

A. Lokshin and M. Zak
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109

JJ574450

It is shown
~~Abstract . This paper shows~~ that a simple two-link planar manipulator exhibits a phenomenon of global instability in a subspace of its configuration space. A numerical example, as well as results of a graphic simulation, is given.

I. Introduction.

The problem of unpredictability in deterministic mechanical systems without random disturbances was posed more than a century ago in connection with turbulence motion. A new interest in the problem was aroused only recently when it became clear that even low order deterministic dynamical systems can be unpredictable from any practical point of view. A classical example of a Henon-Heiles system (1) [2]

$$H = 0.5(p_1{}^2 + q^{12} + p_2{}^2 + q_2{}^2) + q_1{}^2 q_2 - q_3{}^2/3 \qquad (1)$$

represents a case of a well posed deterministic conservative system with only two degrees of freedom. While (1) cannot be solved analytically for arbitrary initial values, it can be integrated numerically. Henon and Heiles did it in 1964, and the results showed that for the system energy above E=1/6, a phase portrait looks seemingly random, while for the E=1/12 <for energy levels above E=1/8, while for E<1/8 system (1) exhibits traditional smooth curves (Fig. 1). During the last twenty years, existence of chaos in nonlinear dynamic systems became a well established fact.

Another type of chaos that is important for us can be well demonstrated by equation (2).

$$X(n+1) = 2 * X(n) \text{ MOD}(1) \qquad (2)$$

For a binary representation of X(n), it simply states that on each step we are shifting the decimal point in X one bit right and throwing out an integer part. It is easy to see that (2) has an analytical solution (3)

$$X(n) = 2^n * X(0) \text{ MOD}(1) \qquad (3)$$

but to compute a result for a given N, one must know exactly N bits in the initial value of X(0).

This example demonstrates a case of orbital instability that was first studied in [1]. Equation (3) is a case in which an initial separation between two close solutions grows exponentially along the trajectories. Of course for any real dynamical system, orbital instability can exist only for the general coordinates that don't increase ths system's total energy. It is worth mentioning here that while an exponential "explosion" of solutions is very well known in the Linear System Theory, there it only means that a linear system description cannot be used, and the system moves toward its limited circle or breaks down as a result of too high stresses. Conversely, an orbital instability in nonlinear systems does not lead to an alternative stable equilibrium, and the system description is done, in the framework of Newtonian mechanics, without any simplification and linearization.

II. Geometric Approach.

For our future discussion, a geometrical representation of orbital instability may be useful. Let us start from an example of an inertial motion of a single particle M on a smooth surface S. In the absence of external forces, a point mass M would move along the geodesics line on this surface. It is shown in differential geometry [5] that the distance between two initially close geodesics is

$$d(t) = d_0 * \exp(t\sqrt{-G}\,) \qquad (4)$$

where d_0 is an initial separation, G - Gaussian curvature, and t is a trajectory parameter not necessarily time. One can see from (4) that for a negative surface curvature the separation increases exponentially. Such a case is shown in Fig. 2. Alternatively, for the surfaces with a positive curvature, separation is bounded by its initial value.

This geometrical representation is very important for a question of system orbital stability. Indeed if it is possible to find a space where the system behavior can be described as an inertial motion of a single particle, one need examine only the sign of a space Gaussian curvature without solving the equations themselves [4].

For the case of conservative finite-degree-of-freedom systems, such a space is very well known. It is a configuration space with a metric tensor corresponding to the structure of the system kinetic energy [4]. Let q^i {i=1..N] denote generalized coordinates, and the kinetic energy is

$$E = a_{ij}\dot{q}^i\dot{q}^j \tag{5}$$

then a_{ij} should be used as a new metric tensor for the configuration space. In (5) and thereafter, summation is assumed upon repeated indexes. In such constructed space, the solution for the free motion of the original system will correspond to an inertial motion of a single particle of a unit mass along the geodesic lines [5,6].

For this metrics triangle, equality is not true any more. Now an elementary arc is

$$ds^2 = a_{ij}dq^idq^j \tag{6}$$

and ds^2 can be less, greater, or equal to the sum of dq_i^2. The sign of the resulting Gaussian curvature is connected to this relation. An illustration of a two dimensional case is shown in Fig. 3.

In the rest of the paper, we are going to show that a free frictionless motion of a simple mechanical system of a two-link planar manipulator, in the absence of gravity, can demonstrate orbital instability that can be characterized as a "weak chaos" [4].

III. Solution for a Two-link Arm.

A model for a two-link planar manipulator is shown in Fig.4. Angles f_1 and f_2 can be chosen as generalized coordinates q_1, q_2. System kinetic energy is

$$E = a_{11}*\dot{f}_1{}^2 + a_{12}*\dot{f}_1*\dot{f}_2 + a_{22}*\dot{f}_2{}^2 \tag{6}$$

$$\begin{aligned} a_{11} &= (I_1 + m*L^2) \\ a_{12} &= m*l*lg*\cos(f_2-f_1) \\ a_{22} &= I_2 \end{aligned} \tag{7}$$

where I_1 and I_2 are moments of inertia, m - mass of the link 2, L - length of link 1, and lg is the distance from B to the center of inertia of the link 2.

The curvature of the resulting two dimensional space can be computed explicitly

$$G*a^2 = m*L*lg*(a^2+[m*L*lg*\sin(f_2-f_1)]^2)^2*\cos(f_2-f_1) \tag{8}$$

and differential equations for f_1 and f_2 can be solved numerically.

One can see from (8) that the sign of the curvature G depends only on the $\cos(f_2-f_1)$. Fig.5 gives the region on (f_1,f_2) plane where G is positive and therefore the system is orbitally unstable. In other words folded arm configurations are orbitally unstable, and extended arm configurations are orbitally stable.

IV. Numerical Simulation.

For a numerical simulation, we chose parameter values that had been used for a real manipulator [7].

$$\begin{aligned} &I_1 = 0.126,\ I_2 = 0.075,\ m = 4.978 \\ &L = 0.27,\ lg = 0.0485 \end{aligned} \tag{9}$$

The system was exercised in the following way. For various initial conditions, two arms were run with a slight difference in their start points. The time history, configuration plane trajectories, as well as graphical animation of the arms themselves had been displayed.

Also we computed and graphically displayed a running estimation of a Lyapunov exponent for the f_2-f_1

$$L(t) = [\ln(d(t)/d_0)]/t \qquad (10)$$

where d(t) is the difference between the arms in the value of an internal angle ($f_2(t)$-$f_1(t)$). It had been shown [8] that a motion is chaotic if

$$L(t) \rightarrow c > 0;\ \text{when } t \rightarrow \inf \qquad (11)$$

Integration was done by using Runge-Kutta with adaptive size steps. To avoid the influence of numerical errors, integration was repeated with a tolerance parameter varying more than an order of magnitude. All runs gave the same results within desired tolerance.

V. Results.

While the fact of positive but not constant curvature over the whole space is a necessary but not sufficient condition for the orbital stability, the opposite is true. A system is orbitally unstable if the curvature of the space defined by (5),(6) is negative in all points.

Since there are "good" and "bad" regions on the f_1 vs f_2 plane (as shown on Fig.5), to get definite results it would be desirable to find an initial condition that would keep the system only in the region with positive or negative curvature. However it is clear, that if one can hope to find a trajectory that stays completely in the "good" region (G>0), there is no trajectory that would stay in the region with orbital instability. Indeed, any solution for the f_2-f_1 is of MOD(2PI), and the separation can not grow indefinitely without forcing an arm to "unfold".

In our simulation we found a case when an arm starting from unfolded position would stay there. In that case the original difference between the arm almost did not grow as can be seen from Fig.6a. This case will be further referred to as Case I. Its initial conditions were :

arm1 : f_1 = 80, f_2 = 95 [dg], $\dot{f}_1$ = 300, $\dot{f}_2$ = 0 [dg/sec]
arm2 : f_1 = 80, f_2 = 97 [dg], $\dot{f}_1$ = 300, $\dot{f}_2$ = 0 [dg/sec]

For the arm started from a folded position a small initial difference grew very fast as could be seen from Fig.6b. Fig.6c shows arms in one of the intermediate positions. It is clear how far apart they become. The initial conditions for the Case II were :

arm1 : f_1 = 80, f_2 = 200 [dg], $\dot{f}_1$ = 300, $\dot{f}_2$ = 0 [dg/sec]
arm2 : f_1 = 80, f_2 = 202 [dg], $\dot{f}_1$ = 300, $\dot{f}_2$ = 0 [dg/sec]

The estimate of the Lyapunov exponent for the Case I clearly goes to zero, while Case II stayed positive at least during the time of observation - Fig.7a,b. It is not clear nevertheless that it will never go to zero, due to the reasons described above (crossing both good as well as bad regions).

VI. Discussion.

A phenomenon of chaotic motion has been theoretically found and numerically illustrated for a simple mechanical system of a two-link manipulator. It has been shown that a folded arm is orbitally unstable, opposite to an extended one.

While the assumption of a nonfriction, zero gravity environment is quite unrealistic, we believe that our finding warrants further and more detailed investigations of the described phenomena. It is quite possible that low-friction, many degrees of freedom redundant flexible arms of the future will exhibit more complicated behavior that could lead to orbital instability under more realistic conditions.

The importance of the geodesics in the configuration space with metric a_{ij} as minimum time trajectories has been recently shown by K.G. Shin and N.D. McKay [9]. Our result suggests that an open loop control should not be used in the region with negative curvature since trajectories there rapidly diverge.

A question of a closed loop control in the area of negative curvature also deserves more detailed investigation. While it had been shown that a simple PID control under the same conditions (no friction and no gravity) makes an arbitrary robot manipulator asymptotically stable [10] it is not clear how trajectory curvature affects an admissible sampling rate.

VII. Conclusion.

Using a geometric approach we have shown that a simple robot manipulator can be orbitally unstable depending on its configuration. A numerical simulation supported this finding. We feel that further efforts in this direction will help better understanding of the dynamical properties of such complicated nonlinear systems as robot manipulators.

Acknowledgement

This work was done at the Jet Propulsion Laboratory, California Institute of Technology under a contract with the National Aeronautics and Space Administration (NASA).

Literature :

[1] W.Thompson,P.G.Tout, "Treatise of Natural Philosophy" vol1, part 1, p.416 1879.

[2] M.Henon, C.Heiles, Astron.J. 69,73 . 1964

[3] Joseph Ford "How random is a coin toss?" Physics Today, April 1983, pp 40-47.

[4] M. Zak "Two Types of Chaos in Non-Linear Mechanics" I.J of Nonlinear Mechanics, v. 20, Nov 4, pp. 297-308, 1985.

[5] J.L. Singe "On the Geometry of Dynamics". Phil. Trans. R. Soc. Lond., Ser A,226, pp.31-106 (1096)

[6] A.I. Lurie "Analytical Mechanics" (Russian) Moscow, 1961

[7] O. Sato,H. Shimojima, Y.Kitamura "Minimal Time Control of a manipulator with 2 degrees of freedom". Bulletin JSME vol 26, no. 218, Aug. 1983

[8] A.J. Lichtenberg, M.A. Lieberman "Regular and Stochastic Motion", Springer-Verlag, NY, 1983

[9] K.G.Shin, N.D. McKay "Selection of Near-minimum Time Geometric Paths for Robotic Manipulators" IEEE Tr. AC v. AC-31 No. 6, June 1986, pp.501-511

[10] M.Takegaki, S.Arimoto "A new Feedback Method for Dynamic Control of Robot Manipulators". JDSMC June 1981, v.102, pp.119-125

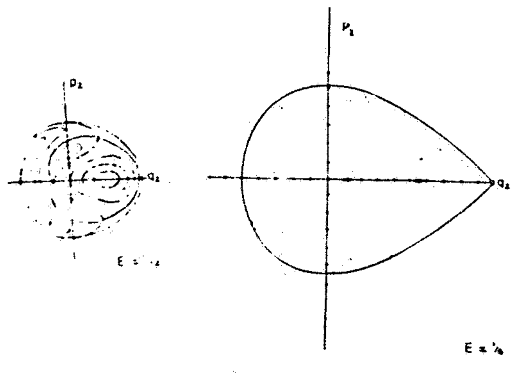


FIGURE 1

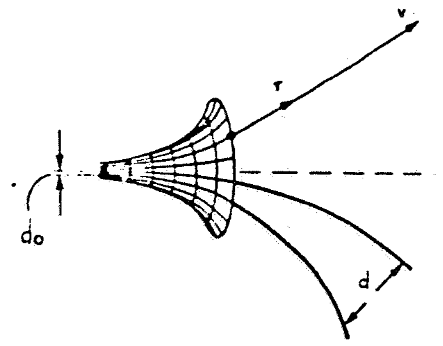


FIGURE 2

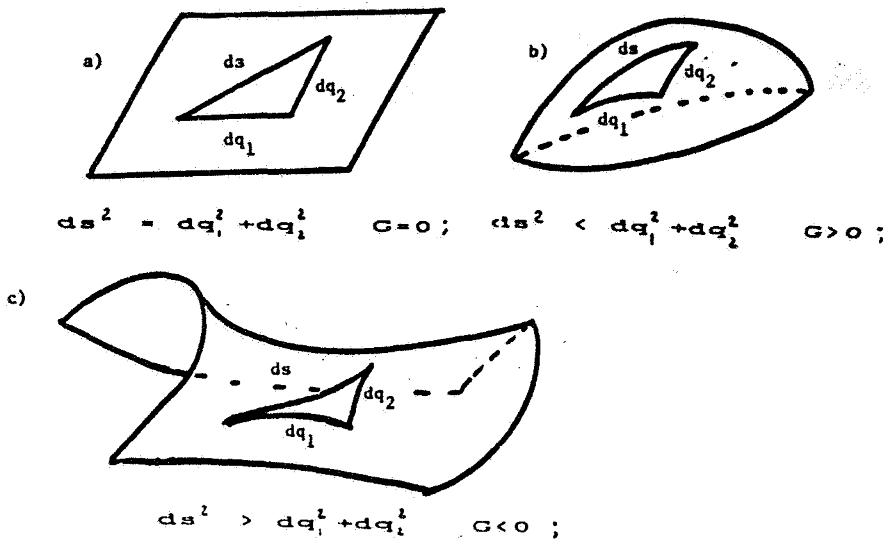

a)
ds
dq_2
dq_1
b)
ds
dq_2
dq_1
$ds^2 = dq_1^2 + dq_2^2 \quad G=0\,;$
$ds^2 < dq_1^2 + dq_2^2 \quad G>0\,;$
c)
ds
dq_2
dq_1
$ds^2 > dq_1^2 + dq_2^2 \quad G<0\,;$

FIGURE 3

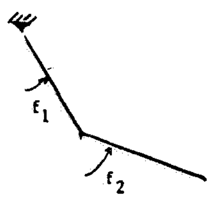


FIGURE 4

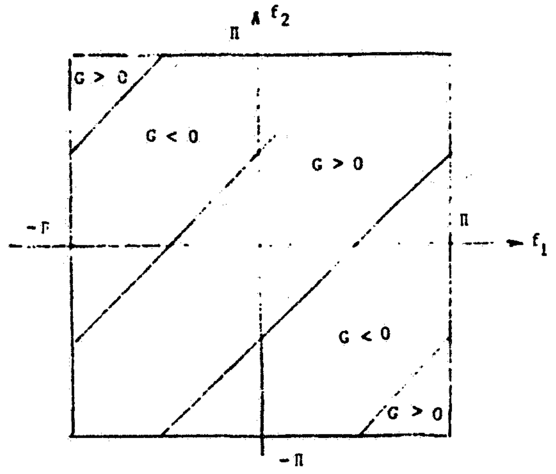
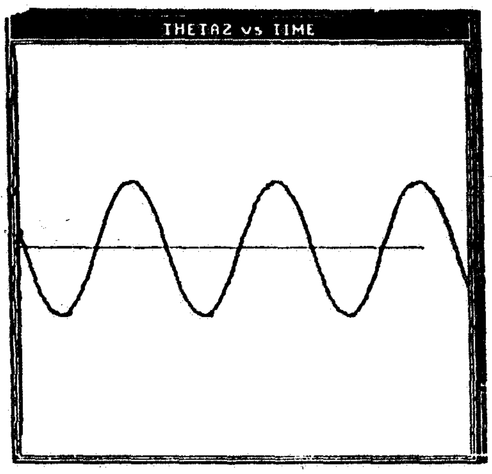
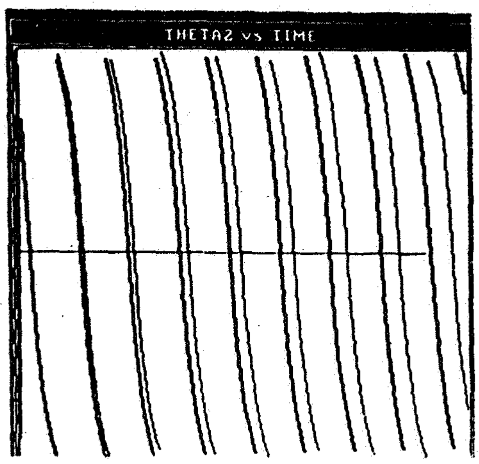


FIGURE 5

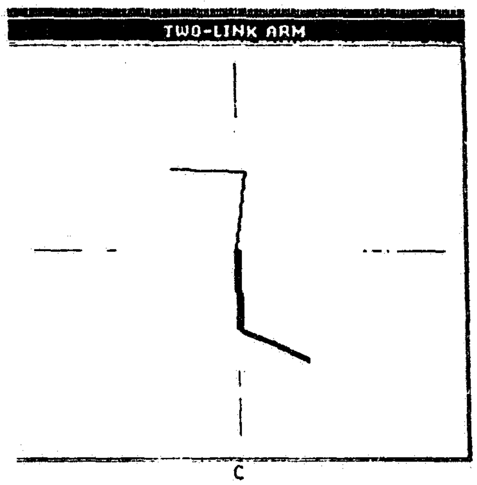


A



B

FIGURE 6



C

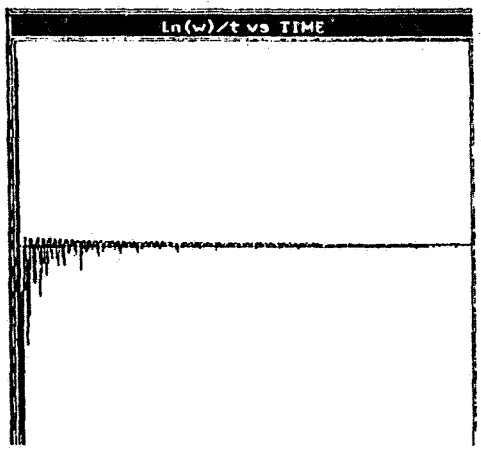


FIGURE 7A

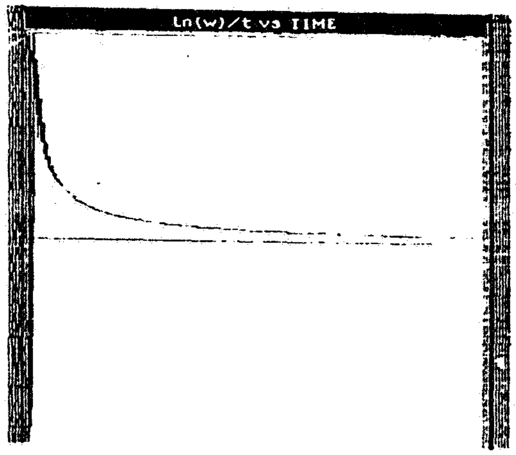


FIGURE 7B

S27-37
9P.
188147

Effect of Control Sampling Rates on Model-Based Manipulator Control Schemes

P.K. Khosla
Carnegie-Mellon University
Pittsburgh, PA 15213

CH 188052

Research

is discussed

Abstract

~~In our previous research, we experimentally implemented and evaluated the effect of dynamics compensation in model-based control algorithms. In this paper, we evaluate~~ the effect of changing the control sampling period on the performance of the computed-torque and independent joint control schemes. While the former utilizes the complete dynamics model of the manipulator, the latter assumes a decoupled and linear model of the manipulator dynamics. We discuss the design of controller gains for both the computed-torque and the independent joint control schemes and establish a framework for comparing their trajectory tracking performance. Our experiments show that within each scheme the trajectory tracking accuracy varies slightly with the change of the sampling rate. However, at low sampling rates the computed-torque scheme outperforms the independent joint control scheme. Based on our experimental results, we also conclusively establish the importance of high sampling rates as they result in an increased stiffness of the system.

research

1. Introduction

Although many simulation results have been presented [13, 12, 4], the real-time implementation and performance of model-based control schemes with high control sampling rates had not been demonstrated on actual manipulators, until recently [9, 11, 1]. The main reasons for this have been the lack of a suitable manipulator system and the fact that it is difficult to evaluate the dynamics parameters for implementing model-based algorithms. One of the goals of the CMU Direct-Drive Arm II [14] project has been to overcome these difficulties and evaluate the effect of dynamics compensation on the real-time trajectory tracking of manipulators. For the real-time computation of the inverse dynamics, we have developed a high-speed and powerful computational environment. The computation of inverse dynamics has been customized for the CMU DD Arm II and a computation time of 1 *ms* has been achieved [5]. To obtain an accurate model we have computed and measured the various parameters from the engineering drawings of the CMU DD Arm II by modeling each link as a composite of hollow and solid cylinders, prisms, and rectangular parallelopipeds. We have also proposed an algorithm to identify the dynamics parameters [8] which has been implemented on the CMU DD Arm II. The results of the experimental implementation of our identification algorithm are presented in [6, 7]. Finally, the negligible friction in our direct-drive arm especially makes it suitable to test the efficacy of the computed-torque scheme.

In our previous research, we investigated the effect of high sampling rate dynamics compensation in model-based manipulator control methods. Specifically, we compared the computed-torque scheme which utilizes the complete dynamics model of the manipulator with the independent joint control scheme [9] and the feedforward compensation method [10]. The control schemes were implemented on the CMU DD Arm II with a sampling period of 2 *ms*. In this paper, we investigate the effect of reducing the sampling rate on the trajectory tracking performance of manipulator control methods. We first compare the performance of each scheme as the sampling rate is changed. Next, we also compare the relative performance of both the computed-torque and the independent joint control schemes at different sampling rates.

This paper is organized as follows: In Section 2, we present an overview of the manipulator control schemes that have been implemented and evaluated on the CMU DD Arm II. The design of controllers is discussed in Section 3 and the real-time experimental results are presented and interpreted in Section 4. Finally, in Section 5 we summarize this paper. In the Appendix, we describe our experimental hardware set-up.

2. Manipulator Control Techniques

The robot control problem revolves around the computation of the actuating joint torques/forces to follow the desired trajectory. The dynamics of a manipulator are described by a set of highly nonlinear and coupled differential equations. The complete dynamic model of an N degrees-of-freedom manipulator is described by:

$$\tau = \mathbf{D}(\theta)\ddot{\theta} + \mathbf{h}(\theta,\dot{\theta}) + \mathbf{g}(\theta) \qquad (1)$$

where τ is the N-vector of the actuating torques; $\mathbf{D}(\theta)$ is the $N \times N$ position dependent manipulator inertia matrix; $\mathbf{h}(\theta,\dot{\theta})$ is the N-vector of Coriolis and centrifugal torques; $\mathbf{g}(\theta)$ is the N-vector of gravitational torques; and $\ddot{\theta}$, $\dot{\theta}$ and θ are N-vectors of the joint accelerations, velocities and positions, respectively.

This complex description of the system makes the design of controllers a difficult task. To circumvent the difficulties the control engineer often assumes a simplified model to proceed with the controller design. Industrial manipulators are usually controlled by conventional PID-type independent joint control structures designed under the assumption that the dynamics of the links are uncoupled and linear. The controllers based on such an overly simplified dynamics model result in low speeds of operation and overshoot of the end-effector.

To establish a framework for comparing the performance these two schemes, we consider the control law in two steps; computation of the commanded acceleration and computation of the control torque. The commanded joint accelerations $\mathbf{u}_i$ can be computed in one of the following three ways:

$$\mathbf{u}_1 = \mathbf{K}_p(\theta_d - \theta) - \mathbf{K}_v\dot{\theta} \qquad (2)$$

$$\mathbf{u}_2 = \mathbf{K}_p(\theta_d - \theta) + \mathbf{K}_v(\dot{\theta}_d - \dot{\theta}) \qquad (3)$$

$$\mathbf{u}_3 = \mathbf{K}_p(\theta_d - \theta) + \mathbf{K}_v(\dot{\theta}_d - \dot{\theta}) + \ddot{\theta}_d \qquad (4)$$

where $\mathbf{K}_p$ and $\mathbf{K}_v$ are $N \times N$ diagonal position and velocity gain matrices, respectively. The N-vectors θ_d and θ are the desired and measured joint positions, respectively, and the " $\cdot$ " indicates the time derivative of the variables. Whereas only the position error and the velocity damping is used in (2), the commanded acceleration signal in (3) uses a velocity feedforward term, and the commanded acceleration signal in (4) uses both the velocity and acceleration feedforward terms. The idea is to increase the speed of response by incorporating a feedforward term.

The fundamental difference between the independent joint control schemes and the model-based schemes lies in the second step in the control law, i.e., the method of computing the applied control torque signals from the commanded acceleration signals. If the vector of actuating joint torques τ is computed from the commanded acceleration signal under the assumption that the joint inertias are constant, then we obtain an independent joint control scheme. On the other hand, if the actuating torques τ are computed from the *inverse dynamics* model in (1) then we obtain the computed-torque scheme.

We have implemented computed-torque and the independent joint control schemes and compared their real-time performance as a function of the sampling rate. These schemes are described in the sequel.

Independent Joint Control (IJC)

In this scheme, linear PD control laws were designed for each joint based on the assumption that the joints are decoupled and linear. The control torque τ applied to the joints at each sampling instant is:

$$\tau = \mathbf{J}\mathbf{u}_i \qquad (5)$$

where $\mathbf{J}$ is the constant $N \times N$ diagonal matrix of link inertias at a typical position.

__Computed-Torque Control (CT)__

This scheme utilizes nonlinear feedback to decouple the manipulator. The control torque τ is computed by the inverse dynamics equation in (1), using the commanded acceleration $\mathbf{u}_t$ instead of the measured acceleration $\ddot{\theta}$:

$$\tau = \tilde{\mathbf{D}}(\theta)\mathbf{u}_t + \tilde{\mathbf{h}}(\theta,\dot{\theta}) + \tilde{\mathbf{g}}(\theta) \qquad (6)$$

where the " ~ " indicates that the estimated values of the dynamics parameters are used in the computation.

The real-time control experiments using these schemes have been performed with the CMU DD Arm II. Also, we have used the Equation 4 to compute the accelerations for both the computed-torque and the independent joint control schemes. Before proceeding with the design of the controller gain matrices, we need to determine the order and transfer function of the individual joint drive systems. We achieved this by performing frequency response experiments. The details of these experiments are presented in [9, 6].

3. Controller Design

The performance of the nonlinear CT scheme and the linear IJC scheme can be compared only if the same criteria are used for design of the controller gain matrices. Fortunately, this is possible because the gain matrices $\mathbf{K}_p$ and $\mathbf{K}_v$ appear only in the commanded accelerations which are the same (Equations (2)-(4)) for both CT and IJC schemes. Thus, whether we implement the simplistic independent joint control scheme or the sophisticated computed-torque scheme, we are faced with the problem of designing the gain matrices $\mathbf{K}_p$ and $\mathbf{K}_v$. These matrices are chosen to satisfy the specified output response criterion.

3.1. Design of Gain Matrices for Independent Joint Control

The closed loop transfer function relating the input θ_{jd} to the measured output θ_j for joint j is:

$$\frac{\theta_j}{\theta_{jd}} = \frac{s^2\delta + s\gamma k_{vj} + k_{pj}}{s^2 + k_{vj}s + k_{pj}} \qquad (7)$$

where $\gamma=1$ if velocity feedforward is included and zero otherwise, and $\delta=1$ if acceleration feedforward is included and zero otherwise. The closed-loop characteristic equation in all the three cases is,

$$s^2 + k_{vj}s + k_{pj} = 0 \qquad (8)$$

and its roots are specified to obtain a stable response. The complete closed-loop response of the system is governed by both the zeros and the poles of the system. In the absence of any feedforward terms, the response is governed by the poles of the transfer function.

Since it is desired that none of the joints overshoot the commanded position or the response be critically damped, our choice of the matrices $\mathbf{K}_p$ and $\mathbf{K}_v$ must be such that their elements satisfy the condition:

$$k_{vj} = 2\sqrt{k_{pj}} \qquad \text{for } j = 1,\ldots\ldots,6 \qquad (9)$$

Besides, in order to achieve a high disturbance rejection ratio or high stiffness it is also necessary to choose the position gain matrix $\mathbf{K}_p$ as large as possible which results in a large $\mathbf{K}_v$.

3.2. Design of Gain Matrices for Computed-Torque Scheme

The basic idea behind the computed torque scheme is to achieve dynamic decoupling of all the joints using nonlinear feedback. If the dynamic model of the manipulator is described by (1) and the applied control torque is computed according to (6), then the following closed-loop system is obtained:

$$\ddot{\theta} = \mathbf{u}_t + [\tilde{\mathbf{D}}]^{-1}\{[\mathbf{D} - \tilde{\mathbf{D}}]\ddot{\theta} + [\mathbf{h} - \tilde{\mathbf{h}}] + [\mathbf{g} - \tilde{\mathbf{g}}]\}$$

where the functional dependencies on θ and $\dot{\theta}$ have been omitted for the sake of clarity. If the dynamics are modeled exactly, that is, $\tilde{\mathbf{D}} = \mathbf{D}$, $\tilde{\mathbf{h}} = \mathbf{h}$ and $\tilde{\mathbf{g}} = \mathbf{g}$, then the decoupled closed loop system is described by

$$\ddot{\theta} = \mathbf{u}_t.$$

Upon substituting the right hand side of either (2), (3) or (4) in the above equation, we obtain the closed-loop input-output transfer function of the system. The closed-loop characteristic equation in all the three cases is:

$$s^2 + k_{vj}s + k_{pj} = 0 \tag{10}$$

where k_{vj} and k_{pj} are the velocity and position gains for the *j-th* joint. Upon comparing (8) and (10), we obtain the relationships

$$k_{pj}^{[CT]} = k_{pj}^{[IJC]} \quad \text{and} \quad k_{vj}^{[CT]} = k_{vj}^{[IJC]}$$

which suggest that the gains of the IJC scheme are also the gains of the CT scheme. This equality must be expected because the closed-loop characteristic equation for both the independent joint control and the computed-torque scheme is the same.

3.3. Gain Selection

The gain matrices K_p and K_v are a function of the sampling rate of the control system [3]. The higher the sampling rate the larger the values of K_p and K_v can be chosen. Since the stiffness (or disturbance rejection property) of the system is governed by the position gain matrix a higher sampling rate implies higher stiffness also. In practice the choice of the velocity gain K_v is limited by the noise present in the velocity measurement. We determined the upper limit of the velocity gain experimentally: we set the position gain to zero and increased the velocity gain of each joint until the unmodeled high-frequency dynamics of the system were excited by the noise introduced in the velocity measurement. This value of K_v represents the maximum allowable velocity gain. We chose 80% of the maximum velocity gain in order to obtain as high value of the position gain as possible and still be well within the stability limits with respect to the unmodeled high frequency dynamics. The elements of the position gain matrix K_p were computed to satisfy the critical damping condition in (9) and also achieved the maximum disturbance rejection ratio. The elements of the velocity and position gain matrices used in the implementation of the control schemes are listed in Table 1.

4. Experiments and Results

4.1. Trajectory Selection and Evaluation Criteria

Since the DD Arm II is a highly nonlinear and coupled system it is impossible to characterize its behavior from a particular class of inputs, unlike linear systems for which a specific input (such as a unit step or a ramp) can be used to design and evaluate the controllers. Thus an important constituent of the experimental evaluation of robot control schemes is the choice of a class of inputs for the robot. The criteria for selecting the joint trajectories is detailed in [6]. For evaluating the performance of robot control schemes, we use the dynamic tracking accuracy. This is defined as the maximum position and velocity tracking error along a specified trajectory.

4.2. Real-Time Results

In our experiments we implemented both the independent joint control scheme and the computed-torque scheme. We evaluated their individual and relative performances by changing the sampling rate but keeping both the position and the velocity gain matrices fixed. The maximum permissible velocity and position gains were chosen at a control sampling period of 5 ms (according to the method outlined in Section 3.3) and remained fixed even when the sampling period was changed. This allows us to determine the effect of the sampling rate on the trajectory tracking control performance. We have also evaluated the best performance of the CT method for a sampling period of 2 ms with its best performance for a sampling period of 5 ms. We conducted the evaluation experiments on a multitude of trajectories but due to space limitations we present our results for a simple but illustrative trajectory.

The first trajectory is chosen to be simple and relatively slow but capable of providing insight into the effect of dynamics compensation. In this trajectory only joint 2 moves while all the other joints are commanded to hold their zero positions and can be envisioned from the schematic diagram in Figure 1. Joint 2 is commanded to start from its zero position and to reach the position of 1.5 rad in 0.75 seconds; it remains at this position for an interval of 0.75 seconds after which it is required to return to its home position in 0.75 seconds. The points of discontinuity, in the trajectory, were joined by a fifth-order polynomial to maintain the continuity of position, velocity and acceleration along the three segments. The desired position, velocity and acceleration trajectories for joint 2 are depicted in Figure 2. The maximum velocity and acceleration to be attained by joint 2 are 2 rad/sec and 6 rad/sec^2, respectively.

The position tracking performance of joint 2 for both the CT and IJC schemes, for a control sampling rate of 200 Hz (corresponding to a control sampling period of 5 ms), is depicted in Figure 3. The corresponding position and velocity tracking errors are presented in Figures 4 and 5, respectively. We also depict the position tracking error of joint 1 in Figure 6 for both the CT and IJC schemes. We note that the CT scheme outperforms the IJC scheme. For example, in the case of joint 2 the maximum position tracking error for CT scheme is 0.03 rads while for the IJC scheme it is 0.45 rads, approximately. In an earlier paper [9], we had compared both the CT and IJC schemes with a control sampling period of 2 ms. It must be noted that in the earlier reported experiments [9] the gains were selected for a control sampling period of 2 ms whereas in the present experiments the gains have been selected for a control sampling period of 5 ms. To put the results in perspective, we recall that in the earlier experiment the maximum position tracking error for the CT method was 0.022 rads while for the IJC method it was 0.036 rads. From the above observations it may be deduced that increasing the control sampling period from 2 to 5 ms results in a noteworthy degradation of the performance of the IJC scheme. A similar increase in the sampling rate also improves the performance of the CT scheme.

In Figure 7, we depict the performance of the CT scheme as the sampling rate is increased from 200 Hz to 500 Hz. In this case the position and velocity gain matrices were determined for a sampling rate of 200 Hz and they remained fixed even when the sampling rate was increased to 500 Hz. Thus, Figure 7 presents the relative performance of the CT method as a function of the sampling rate only. We note that the trajectory tracking performance for both 200 Hz and 500 Hz sampling rates is comparable and has not changed in any appreciable manner with an increase in the sampling rate. Figure 8 depicts the results for the IJC method when a similar experiment was performed. In this case also we do not observe any appreciable change in performance when only the sampling rate is changed.

Thus, from the above set of experiments the following conclusions may be drawn:

1. If the gains are selected for a lower sampling rate and then if the sampling rate is increased, while keeping the gains fixed, there is no appreciable improvement in the performance of both the CT and the IJC schmes.

2. At lower sampling rates the CT scheme outperforms the IJC method. Even though the disturbance rejection ratio of both the schemes is diminished, it does not appreciably affect the CT method because of the compensation for the nonlinear and coupling terms. Whereas it affects the IJC method because the disturbance that is constituted by the nonlinear and the coupling terms is not rejected appreciably.

3. If the maximum possible gains are selected for the chosen sampling rates then the performance of CT at a higher sampling rate is better than its performance at a lower sampling rate. A similar conclusion is drawn for the IJC scheme also.

Our last conslusion is especially significant because it suggests that a higher sampling rate does not only imply improved performance but it also allows us to achieve high stiffness. It is desirable for a manipulator to have high stiffness so that the effect of unpredictable external disturbances on the trajectory tracking performance is significantly reduced.

5. Summary

In this paper, we have presented the first experimental evaluation of the effect of the sampling rate on the performance of both the computed-torque and the independent joint control schemes. We have discussed the design of the controller gains for both the independent joint control and the computed-torque schemes and established a framework for the comparison of their trajectory tracking performance. Based on our experiments we have demonstrated that the computed-torque scheme exhibits a better performance than the independent joint control scheme. Our experiments also show that high sampling rates are important because they result in a stiffer system that is capable of effectively rejecting unknown external disturbances.

6. Acknowledgements

This research was supported in part by the National Scienece Foundation under Grant ECS-8320364 and the Department of Electrical and Computer Engineering , Carnegie Mellon University. The author acknowledges the cooperation of Prof. Takeo Kanade (Head of Vision Laboratory, Carnegie Mellon University) throughout the course of this research.

I. The CMU DD Arm II

We have developed, at CMU, the concept of direct-drive robots in which the links are directly coupled to the motor shaft. This construction eliminates undesirable properties like friction and gear backlash. The CMU DD Arm II [14] is the second version of the CMU direct-drive manipulator and is designed to be faster, lighter and more accurate than its predecessor CMU DD Arm I [2]. We have used brushless rare-earth magnet DC torque motors driven by current controlled amplifiers to achieve a torque controlled joint drive system. The SCARA-type configuration of the arm reduces the the torque requirements of the first two joints and also simplifies the dynamic model of the arm. To achieve the desired accuracy, we use very high precision (16 bits/rotation) rotary absolute encoders. The arm weighs approximately 70 pounds and is designed to achieve maximum joint accelerations of 10 rad/sec^2.

The hardware of the DD Arm II control system consists of three integral components: the Motorola M68000 microcomputer, the Marinco processor and the TMS-320 microprocessor-based individual joint controllers. We have also developed the customized Newton-Euler equations for the CMU DD Arm II and achieved a computation time of 1 *ms* by implementing these on the Marinco processor. The details of the customized algorithm, hardware configuration and the numerical values of the dynamics parameters are presented in [5].

Joint (j)	Transfer Function ($\frac{1}{J_j s^2}$)	k_{pj}	k_{vj}
1	$\frac{1}{12.3s^2}$	2.75	3.33
2	$\frac{1}{2s^2}$	15.0	7.5
3	$\frac{1}{0.25s^2}$	256.0	32.0
4	$\frac{1}{0.007s^2}$	1285.0	71.5
5	$\frac{1}{0.006s^2}$	625.0	50.0
6	$\frac{1}{0.0003s^2}$	1110.0	50.0

Table 1: *Transfer Functions and Gains of Individual Links*

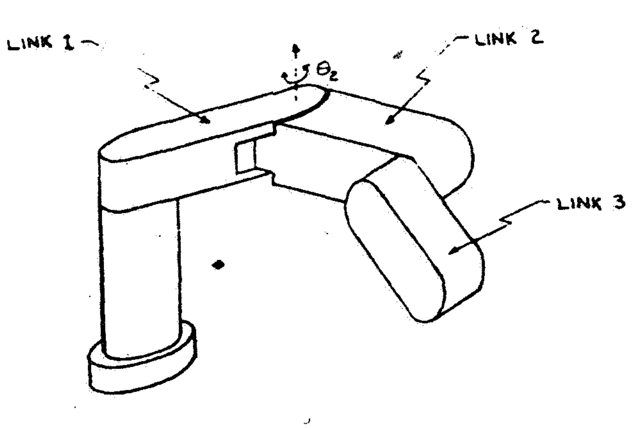


Figure 1: Schematic Diagram of 3 DOF DD Arm II

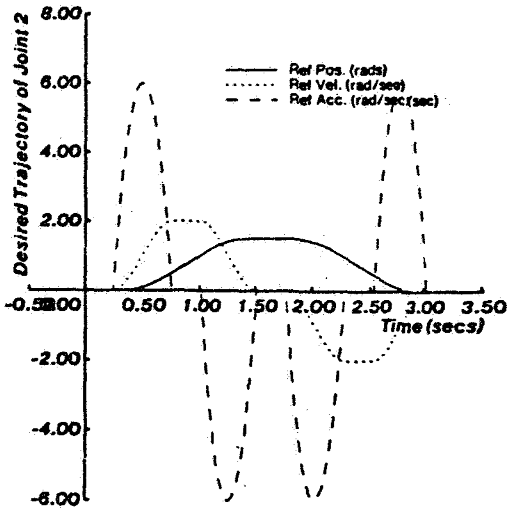


Figure 2: Desired Trajectories for Joint 2

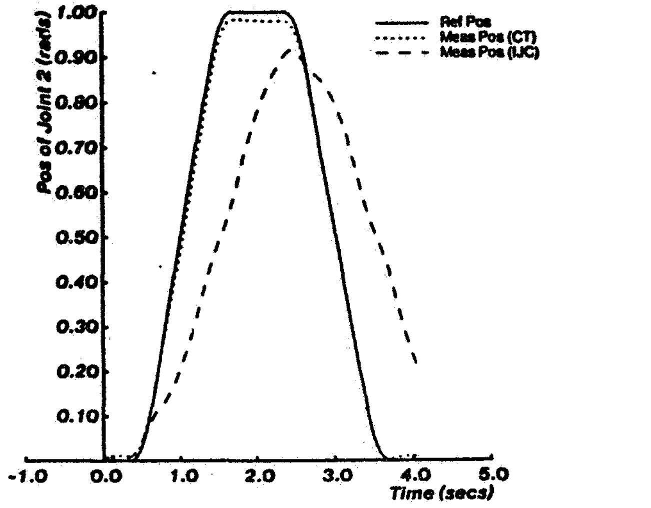


Figure 3: Position Tracking of CT and IJC at 5 ms Sampling

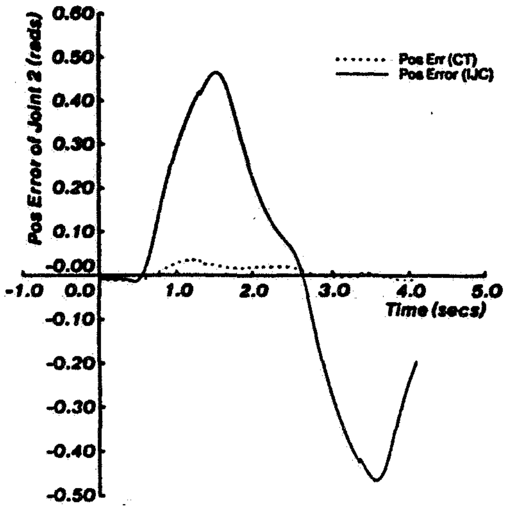


Figure 4: Position Tracking Error of Joint 2

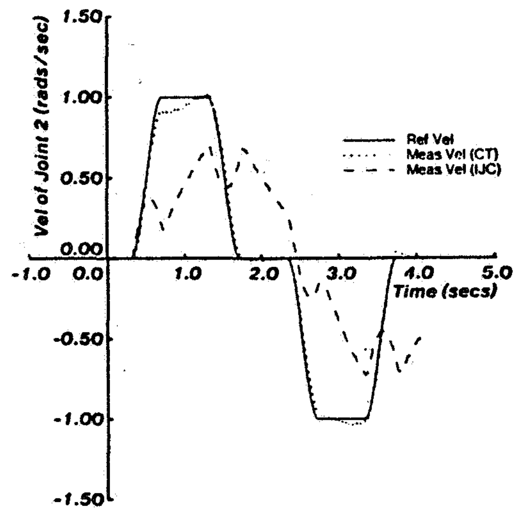


Figure 5: Velocity Tracking Errors of Joint 2

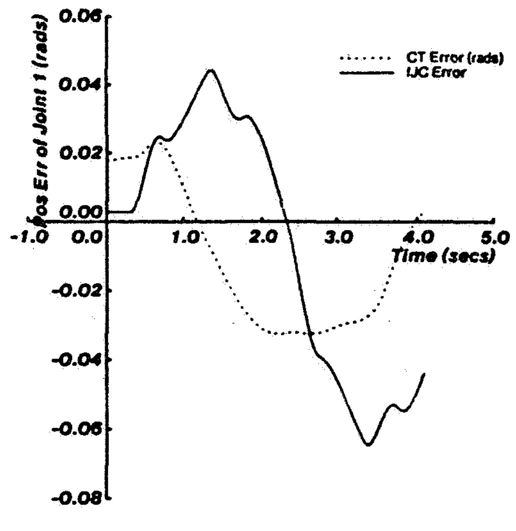


Figure 6: Position Tracking Errors of Joint 1

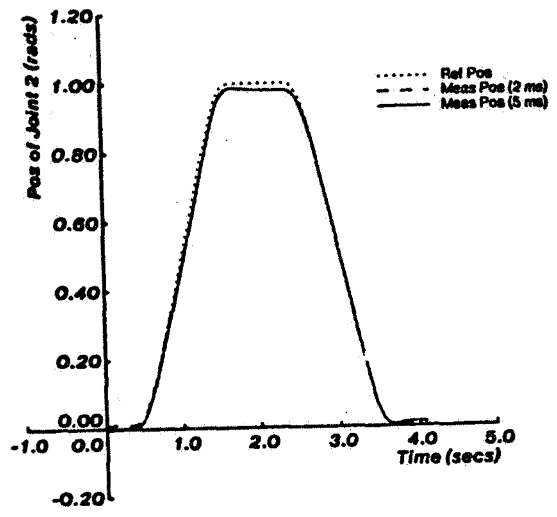


Figure 7: Performance of CT as a Function of Sampling Period

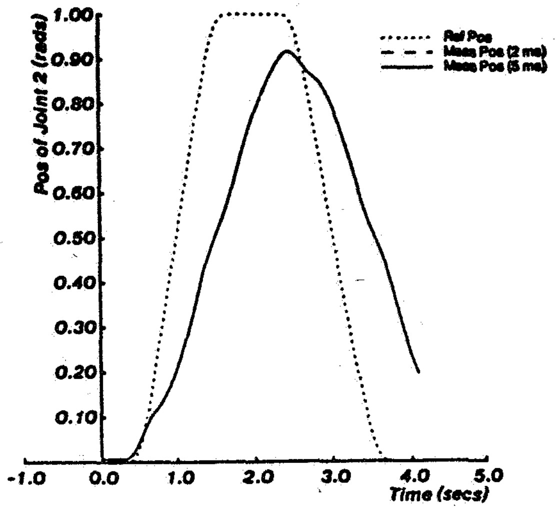


Figure 8: Performance of IJC as a Function of Sampling Period

References

[1] An, C. H., Atkeson, C. G. and Hollerbach, J. M.
Experimental Determination of the Effect of Feedforward Control on Trajectory Tracking Errors.
In Bejczy, A. K. (editor), *Proceedings of 1986 IEEE Conference on Robotics and Automation*, pages 55-60. IEEE, San Francisco, CA, April 7-10, 1986.

[2] Asada, H. and Kanade, T.
Design of Direct Drive Mechanical Arms.
Journal of Vibration, Stress, and Reliability in Design 105(1):312-316, July, 1983.

[3] Astrom, K. J. and Wittenmark, B.
Information and System Science Series: Computer Controlled Systems: Theory and Design.
Prentice-Hall, Englewood Cliffs, N. J., 1984.

[4] Bejczy A. K.
Robot Arm Dynamics and Control.
Technical Memorandum 33-669, Jet Propulsion Laboratory, Pasadena, CA, February, 1974.

[5] Kanade, T., Khosla, P. K. and Tanaka, N.
Real-Time Control of the CMU Direct Drive Arm II Using Customized Inverse Dynamics.
In Polis, M. P. (editor), *Proceedings of the 23rd IEEE Conference on Decision and Control*, pages 1345-1352. Las Vegas, NV, December 12-14,, 1984.

[6] Khosla, P. K.
Real-Time Control and Identification of Direct-Drive Manipulators.
PhD thesis, Department of Electrical and Computer Engineering, Carnegie-Mellon University, August, 1986.

[7] Khosla, P. K.
Estimation of Robot Dynamics Parameters: Theory and Application.
In *Proceedings of the Second International IASTED Conference on Applied Control and Identification*. ACTA Press, Los Angeles, CA, December 10-12, 1986.

[8] Khosla, P. K. and Kanade, T.
Parameter Identification of Robot Dynamics.
In Franklin, G. F. (editor), *Proceedings of the 24-th CDC*, pages 1754-1760. Florida, December 11-13, 1985.

[9] Khosla, P. K. and Kanade, T.
Real-Time Implementation and Evaluation of Model-Based Controls on CMU DD ARM II.
In Bejczy, A. K. (editor), *1986 IEEE International Conference on Robotics and Automation*. IEEE, April 7-10, 1986.

[10] Khosla, P. K. and Kanade, T.
Experimental Evaluation of the Feedforward Compensation and Computed-Torque Control Schemes.
In Stear, E. B. (editor), *Proceedings of the 1986 ACC*. AAAC, Seattle, WA, June 18-20, 1986.

[11] Leahy, M. B., Valavanis, K. P. and Saridis, G. N.
The Effects of Dynamics Models on Robot Control.
In *Proceedings of the 1986 IEEE Conference on Robotics and Automation*. IEEE, San Francisco, CA, April, 1986.

[12] Luh, J. Y. S., Walker, M. W. and Paul, R. P.
Resolved-Acceleration Control of Mechanical Manipulators.
IEEE Transactions on Automatic Control 25(3):468-474, June, 1980.

[13] Markiewicz, B. R.
Analysis of the Computed-Torque Drive Method and Comparision with the Conventional Position Servo for a Computer-Controlled Manipulator.
Technical Memorandum 33-601, Jet Propulsion Laboratory, Pasadena, CA, March, 1973.

[14] Schmitz, D., Khosla, P. K. and Kanade, T.
Development of CMU Direct-Drive Arm II.
In Hasegawa, Yukio (editor), *Proceedings of the 15-th International Symposium on Industrial Robotics*. Tokyo, Japan, September, 11-13, 1985.

S28-37
118
188148

Kinematically Redundant Robot Manipulators*

J. Baillieul
Scientific Systems, Inc. and Boston University
Cambridge, MA 02140
SE 297506
BU 370927

R. Brockett
Harvard University
Cambridge, MA 02138
HH 365172

J. Hollerbach
Massachusetts Institute of Technology
Cambridge, MA 02139
MJ 700802

D. Martin, R. Percy, and R. Thomas
Scientific Systems, Inc.
Cambridge, MA 02140 SE 297506

1.0 Introduction

~~This paper reports~~ research on control, design and programming of kinematically redundant robot manipulators (KRRM) is discussed. These are devices in which there are more joint space degrees of freedom than are required to achieve every position and orientation of the end-effector necessary for a given task in a given workspace. The technological developments described ~~in this paper~~ here deal with:

- Kinematic programming techniques for automatically generating joint-space trajectories to execute prescribed tasks;
- Control of redundant manipulators to optimize dynamic criteria (e.g., applications of forces and moments at the end-effector that optimally distribute the loading of actuators); and
- Design of KRRMs to optimize functionality in congested work environments or to achieve other goals unattainable with non-redundant manipulators.

~~We discuss~~ kinematic programming techniques are discussed, which show ~~showing~~ that some pseudo-inverse techniques that have been proposed for redundant manipulator control fail to achieve the goals of avoiding kinematic singularities and also generating closed joint-space paths corresponding to close paths of the end effector in the workspace. The extended Jacobian is proposed as an alternative to pseudo-inverse techniques. It incorporates functional constraints in a straightforward way to resolve redundancy, and can meet a variety of spatially-varying optimality criteria. This method can generate manipulator trajectories that automatically avoid obstacles provided suitable distance functions are defined, and if the intersections of the constraint surfaces are characterized in a sufficiently simple way.

2.0 Design Issues

A six degree-of-freedom geometry can no longer be considered a general purpose manipulator. This geometry has fatal kinematic flaws that arise from singularities and restrictions on the workspace. The major flaw of six degree-of-freedom manipulators is the presence of singularities in the interior of the workspace. It is exceedingly difficult to plan trajectories that do not pass through or near singularities, given the complex transformation between end effector locations and joint angles. An extra degree of freedom makes functional interior workspace points in the sense that a nonlinear configuration can be found that will correspond to a given workspace point. Singular configurations will still arise, but they can be avoided through exercise of a self-motion to arrive at a new configuration. A self-motion is created by a redundancy and is defined as an internal motion of the linkage that does not move the endpoint. The trajectory planner must still be wary of interior singularities, but upon arriving at one, the motion can backtrack so as to evolve to a different configuration at the singular point. Thus a seven degree-of-freedom represents a minimal configuration (least complex geometry) that makes available all interior workspace points.

Seven degree-of-freedom geometries are complex and costly. Most industry efforts have therefore focused on seeking methods to mitigate the effects of singularities. Strict realization of the velocity requirements at the endpoint must be abandoned. Sometimes a self-motion at the singularity can be used to find an alternative configuration for which the possible endpoint velocities happen to coincide with the desired one [1], although the manipulator must effectively come to a stop for this self-motion to occur.

3.0 Resolution of Redundancy

Redundancy resolution schemes fall into two broad categories: local optimization or global optimization techniques. Within each category, the optimization may be done at the kinematic or at the dynamic level.

*Work conducted under Contract Number F33615-84-C-5131 SBIR Phase II, USAF AFWAL/MLTC

PAGE 244 INTENTIONALLY BLANK

Most research has involved the instantaneous or local resolution of the redundancy through use of the pseudo-invers These local techniques deal with the instantaneous kinematics of motion, *i.e.*, motion which is locally optimized by increme tal movement from the current arm state.

Global optimization minimizes some performance index across a whole trajectory, and hence should perform bett than local optimization. Yet the complexity of problem formulation and the computational interactibility have restricted tl use of global optimization schemes for redundant manipulators.

The advantages of the local optimization methods over global methods are twofold: the simplicity of problem formul tion and the relatively small amount of computation required for the algorithm. The small amount of computation associat with local methods offers the possibility of real-time control of the manipulators. The local technique, however, may n always be desirable for controlling redundant arms. [2] showed motions of a redundant manipulator following closed ha trajectories are generally not closed in joint space trajectories. [3] proved that, without a modification, the generalized inver method need not even avoid kinematically singular configurations. Since the local optimization method only instantaneou minimizes a given criterion, it does not guarantee a global minimum and may even result in a disastrous manipulator moti [4].

On the other hand, the global optimization technique ensures a solution with a global minimum. Real-time contı based on global techniques is problematic, due to the heavy computational requirements. The global technique may be pı fectly adequate for commonly encountered industrial problems requiring repetitive motion, since a specific solution will used over and over again.

3.1 Local Kinematic Resolution of Redundancy

Most local kinematic techniques resolve redundancy at the velocity level by using the pseudo-inverse J^T (also known the Moore Penrose generalized inverse) of the Jacobian J:

$$\dot{x} = J\dot{\theta}$$

$$\dot{\theta} = J^{*} \dot{x} + (I - J^{*}J)\, \dot{\phi}$$

$$J^{T} = J^{T}(JJ^{T})^{-1}$$

where

$\dot{x}=$	6 dimensional velocity vector of the manipulator end
$\dot{\theta}=$	$n > 6$ dimensional joint angle vector
$\dot{\phi}=$	arbitrary joint vector

$(I - J^{T}J)\, \dot{\phi}$ is the projection of $\dot{\phi}$ into the null space of jacob and corresponds to self-motion of the linkage that does ı move the end effector.

This approach is attractive in two ways. First, the pseudo-inverse has a least squares property that can minimize exc sive joint velocities and make smoother motion. Second, the redundancy that is available is succinctly characterized by t null-space of the Jacobian. Measures related to this formulation can be used to achieve some objective, *i.e.*, to avoid jo limits, singularities and obstacles [5,6,7]. A weighted pseudo-inverse (different from the null-space vector) can be used angers high and low priority of variables [8].

The Moore-Penrose generalized invoice is problematic, however, in that it is nonconservative [2]. Repetitive motic planned with the pseudo-inverse alone need not follow a repetitive path in joint-space.

3.2 Global Kinematic Resolution of Redundancy

Nakamura [11] presented a method based on Pontryagin's Maximum Principle for globally optimizing a given cost fu tion for problems involving both kinematics and dynamics. An integral performance index of the following type is minimiz over a desired trajectory:

$$\int_{t_i}^{t_f} \underline{p}(\, \underline{\theta}, t)\, dt$$

where t_i and t_f are the initial and final time respectively. For example, $\underline{p} = \dot{\theta}^{T}\dot{\theta} + kw$, where k is a constant and w is the ma pulatability index, was used by [11]. Pontryagin's Maximum Principle is then applied to Equation 4 and Equation 2 whicl treated as an ordinary optimal control problem of a dynamic system with $\underline{\dot{\theta}}$ as an input vector. The Hamiltonian according a fixed time problem with a fixed time problem with a fixed left hand end-point and a free right hand endpoint is given by

$$H(\underline{\psi}, \underline{\theta}, t, \dot{\underline{\phi}}) = -p + \underline{\psi}^T\dot{\underline{\theta}} \tag{5}$$

where $\underline{\psi}$ is an auxiliary variable vector. The global solution is then given by choosing a $\dot{\underline{\phi}}$ that maximizes the Hamiltonian at every instant and solving the following 2n differential equations:

$$\dot{\underline{\theta}} = \left(\frac{\partial H}{\partial \underline{\psi}}\right)^T \tag{6}$$

$$\dot{\underline{\psi}} = -\left(\frac{\partial H}{\partial \underline{\theta}}\right)^T \tag{7}$$

where Equation 6 is the same as Equation 2.

3.3 Global Kinetic Resolution of Redundancy

For problems including dynamics, a state vector $\underline{\nu} = [\underline{\theta}^T\dot{\underline{\theta}}^T]^T$ was introduced in [11]. Using the inverse kinematics at the acceleration level, the kinematics equations are rewritten in the following form:

$$\dot{\underline{\nu}} = \underline{Q}(\underline{\nu}, t) + \underline{R}(\underline{\nu})\ddot{\underline{\phi}} \tag{8}$$

$$\underline{Q}(\underline{\nu}, t) = \begin{bmatrix} \dot{\underline{\theta}} \\ J\dagger(\ddot{x}(t) - \dot{J}\dot{\underline{\theta}}) \end{bmatrix} \tag{9}$$

$$\underline{R}(\underline{\nu}) = \begin{bmatrix} \underline{0} \\ I - J\dagger J \end{bmatrix} \tag{10}$$

Joint torques can now be written in terms of $\underline{\nu}$, $\ddot{\underline{\phi}}$, and t as

$$\underline{\tau}(\underline{\nu}, \ddot{\underline{\phi}}, t) = \underline{U}(\underline{\nu}, t) + \underline{V}(\underline{\nu})\ddot{\underline{\phi}} \tag{11}$$

$$\underline{U}, (\underline{\nu}, t) = HJ\dagger(\ddot{\underline{x}}(t) - \dot{J}\dot{\underline{\theta}}) + \dot{\underline{\theta}}.C.\dot{\underline{\theta}} + \underline{g} \tag{12}$$

$$\underline{V}(\underline{\nu}) = H(I - J\dagger J) \tag{13}$$

An integral performance index of the following type is then minimized:

$$\int_{t_i}^{t_f} (kp_o(\underline{\nu}) + \underline{\tau}^T\underline{\tau})dt \tag{14}$$

where k is a non-negative scalar. For example, setting k to 0 minimizes the joint torques in a least squares sense. The optimization problem can be solved through Pontryagin's Maximum Principle. The solution requires solving 4n differential equations. The algorithms used in Nakamura's dynamic method and the global algorithm presented in this paper are theoretically equivalent, but different methods are used in the formulation.

4.0 Kinematic Programming Techniques

4.1 Pseudo-Inverse Techniques for Redundancy Resolution

The practical problem associated with planning joint-space motions for kinematically redundant manipulators is that of producing an arbitrary prescribed end-effector movement. To do so, the controller must choose among infinitely many corresponding joint space movements.

For any robot, each possible joint angle configuration defines a unique position of the end effector of the robot arm. This is expressed mathematically by an equation of the form $f(\theta) = x$, where x is a vector (typically six dimensional) defining the position and orientation of the end effector, and θ is a vector defining the joint angle configuration. By differentiating both sides of the equation $x(t) = f(\theta(t))$, we obtain the kinematic relation

$$\dot{x}(t) = \frac{\partial f}{\partial \theta}(\theta(t))\ \dot{\theta}(t) \tag{15}$$

from which we can compute $\dot{\theta}(t)$ in terms of the prescribed end effector trajectory x(t). One way to uniquely specify a joint velocity vector for each $\dot{x}(t)$ is to use the Moore-Penrose inverse given by

$$\dot{\theta}_0(t) = \frac{\partial f}{\partial \theta}(\theta(t))^+ \dot{x}(t), \tag{16}$$

The joint velocities are minimized by this technique. But since joint velocities can become arbitrarily large near singular configurations [13], this technique appears to show promise for generating joint angle trajectories that automatically avoid singular configurations. However, analysis shows the Moore-Penrose inverse technique, without further restrictions, may generate trajectories which pass arbitrarily close to singular points in joint angle space. Thus singularities are not avoided in any practical sense. This result is in contrast to some claims that have been made in literature [2].

Modifications to the Moore-Penrose pseudo-inverse technique can be made to avoid singularities. An alternative to Equation 16 for defining joint angle trajectories uses a projection operator onto the null space:

$$\dot{\theta} += \frac{\partial f}{\partial \theta}(\theta)^+ \dot{x} + \left[I - \frac{\partial f}{\partial \theta}(0)^+ \frac{\partial f}{\partial \theta}(\theta) \right] \nu \tag{17}$$

ν is a (time varying) vector of the same dimension as θ which remains to be specified. This modification of the Moore-Penrose pseudo-inverse technique can generate trajectories which avoid singular configurations by appropriate choice of $\nu(.)$ in Equation (6).

4.1.1 Functional Constraints for Redundancy Resolutions

A second class of methods for resolving redundancy, quite distinct from the generalized inverse methods, is that of imposing differentiable (for smooth motion) functional constraint relationships on the joint angles:

$$\begin{gathered} \phi_r(\theta_1, \theta_2, \ldots, \theta_k) = 0 \\ \cdot \\ \cdot \\ \cdot \\ \phi_r(\theta_1, \theta_2, \ldots, \theta_k) = 0 \end{gathered} \tag{18}$$

In general, however, it might not be possible to choose ϕ so that $(\theta_1, \theta_2, \theta_3)$ satisfy the redundancy condition $\phi(\theta_1, \theta_2, \theta_3) = 0$ and depend continuously on the coordinates (x, y) of the end effector (a 2-d example of the method using a 3-bar resolute joint, linkage in the plane). It is possible to find ϕ if some arbitrarily small area A of the workspace is excluded from the conditicns, hence resolving the redundancy in a continuous way.

4.1.2 Obstacle Avoidance

An optimality criterion defined in terms of a distance function will depend on how obstacles are represented. A simple way of representing manipulator links is to model them as live segments between adjacent joint coordinate systems. Obstructions in the workspace (modeled as, *e.g.*, primitives) can then be classified according to how and which links in the mechanism can be impeded. Analysis of various geometries will then indicate the cases in which the relative dimensions of the links represent undesirable designs.

There are two major issues in incorporating considerations of obstacle avoidance into the design of kinematically redundant manipulators. First, the basic geometry of the mechanism must be specified. Then, dimensions of the manipulator must be chosen to maximize some measure of its capacity to function in a congested workspace.

Each basic manipulator geometry will require specification of a figure-of-merit. One example of such a figure of merit could be the distance a manipulator could reach behind an obstacle in the workspace, or the area excluded from the workspace because of the obstacle. These figures can be based on manipulator characteristics, workspace and obstacle dimensions, or, if this is not known at the design stage, probabilistic models or parametric analyses.

4.2 Global Optimization Techniques

Our research developed practical numerical methods for resolving redundancy and solving the inverse kinematics problem, by minimizing a global (path integral) velocity criterion. These techniques are of interest because of the form in which the solutions are expressed is similar to that of the pseudo-inverse or Extended Jacobian techniques. This can be contrasted with other numerical techniques in which a repetitive and computationally costly process is used until the solution converges.

a nominal solution is assumed, the problem is linearized, the linear optimal solution is found by a backward and forward sweep, and the linear optimal solution is used to update the nominal solution.

Our method differs from these other numerical techniques:

(1) No approximations or linearizations are required;

(2) The solution is always in the form of a differential equation whose solution is always a feasible joint space trajectory;

(3) The "optimal" solution is found by searching over a relatively small number of parameters comprising the initial conditions of the differential equation; and

(4) The computational requirements of the solution for a particular set of initial conditions are comparable to those of the pseudo-inverse or Extended Jacobian techniques.

Our approach is to view this problem as a boundary value problem (the theoretical basis for this approach is due to Nakamura [11]. We choose to use the additional freedom to minimize an integral of the joint velocities over the path:

$$\text{Minimize} \int_0^T |\dot{\theta}(t)|^2 \, dt \tag{19}$$

subject to the constraint

$$x(t) = f(\theta(t)). \tag{19a}$$

The constraint expresses the requirement the end effector follow a prescribed path in space. It is also possible to express the constraints in terms of velocities:

$$\dot{x}(t) = \frac{\partial f}{\partial \theta}\dot{\theta}(t) = J\dot{\theta}(t) \tag{20}$$

Solutions to the problem Equation 19 are obtained from use of undetermined Lagrange multipliers and the Euler-Lagrange equations, and Equation 19 becomes

$$\text{Minimize } 0 = \int_0^T L(\theta, \dot{\theta}, \lambda)\frac{dt}{\theta}, \lambda \tag{21}$$

with

$$\frac{\partial L}{\partial \theta} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) = 0, \; \frac{\partial L}{\partial \lambda} = 0 \tag{22}$$

This leads to

$$J^T\lambda - \ddot{\theta} = 0 \tag{23a}$$

and

$$f(\theta) - x = 0 \tag{23b}$$

For a kinematically redundant manipulator, the dimensions of J as such Equation 23 overdetermines λ in terms of $\ddot{\theta}$. A direct consequence of this is the relation

$$n_J^T\ddot{\theta} = 0 \tag{24}$$

where n_J is any nullspace vector of J(*i.e.*, $Jn_J = 0$, $n_J^T n_J \neq 0$). Equation 24 is the necessary condition that was sought for joint space trajectories that extremize the integral of Equation (18). A solution for λ that is consistent with Equation 24 is $\lambda = (JJ^T)^{-1}J\ \ddot{\theta}$. When we substitute this solution back into Equation 20, we have

$$(J^T(JJ^T)^{-1}J - I)\ddot{\theta} = 0 \tag{25}$$

or, equivalently, $-P_J\ddot{\theta} = 0$ where P_J is the nullspace projection operator for J. Equations 24 and 25 are equivalent when $(JJ^T)^{-1}$ exists.

Equation 24 provides a second order differential equation that requires two boundary conditions to provide a particular solution.

Analysis of this case where $\phi(0)$ and $\theta(t)$ can vary, but are subject to kinematic constraints at the endpoints, leads to the consequence

$$n_j(\theta(0))^T\dot{\theta}(0) = 0$$
$$n_j(\theta(T))^T\dot{\theta}(T) = 0 \qquad (26)$$

This is the simple statement that, when the only boundary condition on θ is the kinematic relationship, $f(\theta) = x$, then a necessary condition for the cost to be at an extremum is the component of initial and final velocity in the nullspace of J be zero.

4.2.1 Differential Equations Describing Optimal Solutions

The equations of constraint, together with the results of the Euler-Lagrange equations just presented, can be used to derive differential equations for propagating the optimal $\theta(t)$. Such solutions must simultaneously satisfy Equation 21 and the kinematic constraint, $f(\theta) = x$. We have evaluated three ways to obtain differential equations for θ that meet these conditions. They differ in the implied computational requirements and some of the techniques introduce "removable" singularities to the computation of the solution. When singular behavior is not evident, all of the techniques provide the same solution to equivalent boundary value problems. Finally, it should be emphasized these differential equations are necessary but not sufficient for an optimal value of σ in Equation 18.

4.2.2 Direct Solution

The most direct way to obtain a second order differential equation meeting the criteria listed above is to differentiate the constraint equations twice with respect to time, to obtain

$$\ddot{x} = J\ddot{\theta} + \dot{J}\theta dot, \qquad (27)$$

When the pseudo-inverse solution for $\ddot{\theta}$ in terms of $\ddot{x}$ and $\dot{\theta}$ is examined,

$$\ddot{\theta} = J\dagger\left(\ddot{x} - \dot{J}\dot{\theta}\right) \qquad (28)$$

where $J\dagger = J^T(JJ^T)^{-1}$, can one observe that this solution to Equation (27) also satisfies Equation (24), since $n_j^T J\dagger = 0$. This means a joint space trajectory integrated from Equation (28) and appropriate boundary conditions will meet the necessary condition for optimality. Note that, for this resolution to exist, $(JJ^T)^{-1}$ must exist everywhere along the trajectory. This is the equivalent to the requirement there be no kinematic singularities on the trajectory. This does not mean optimal trajectories do not include singularities; it is possible to specify boundary conditions, for example, that are kinematically singular. "Optimal" solutions for such problems exist, but they are not a consequence of Equation (24) or (27).

4.2.3 Reduced Order Solutions

In order to obtain solutions to Equation (28) one must integrate a second order differential equation in a number of variables equal to the dimension of the joint angle vector. In principle, not all of these quantities need to be integrated, as some of them are already determined by the restraint of the kinematic relationship. There are two approaches that take advantage of this situation. The first approach introduces a parameter used to resolve the redundancy explicitly. The second approach uses the nullspace velocity as its parameter. In the latter case, the parameter is not obviously related to the configuration of the manipulator at a particular time, but offers the advantage of introducing no removable or extraneous singularities in the differential equation. In the manipulators examined so far, the number of redundant degrees of freedom is one, but all methods presented can be extended to the case of multiple degrees if freedom.

Both techniques can be derived in precisely the same way, and differ only in the particular functional relationship used to resolve the redundancy.

4.2.3.1 The Reduction Resolution Technique

In the redundancy resolution (RR) technique, a redundancy resolution parameter, $\phi = l(\theta)$, is introduced to resolve the ambiguity remaining after the constraint $f(\theta) = x$ is met.

Specifying both x and ϕ should provide enough information to compute θ. A velocity relationship can be obtained by differentiation:

$$\dot{\phi} = \partial\frac{l}{\partial}\theta\dot{\theta} = m^T\dot{\theta} \qquad (29)$$

In the null space velocity approach, the additional equation is defined directly in terms of the nullspace velocity component,

$$\dot{\alpha} = n_J^T \dot{\theta} \tag{30}$$

These two equations have the same form, and the analysis of each is similar, with substitution of appropriate parameters as required.

Applying kinematic constraints on joint velocity, and solving the resulting set of equations, we obtain a second-order differential equation in a scalar parameter that represents either ϕ or α. The inverse of Extended Jacobian can provide an explicit relationship for $\dot{\theta}$ in terms of $\dot{x}$ and this scalar parameter so that the two together provide the reduced order. If n is the dimension of θ, and n-1 the dimension of x, the two relationships comprise n+2 coupled, first order nonlinear differential equations that must be integrated. This can be compared with Equation (28), which is equivalent to $2n$ *differential equations.*

The principle advantage of the RR approach is that ϕ is simply related to the configuration of the manipulator, and can be found directly. The principle disadvantage of the RR approach is that many "optimal" trajectories, depending on the particular conditions or boundary values, encounter singularities under certain conditions of the parameters. The Extended Jacobian technique removes this singularity algebraically and there is, then, the possibility further work with the RR technique can eliminate this disadvantage.

4.2.3.2 The Nullspace Velocity (NV) Technique

The alternative technique to the RR technique just described is the resolution of the redundancy by a velocity constraint, in particular, the specification of the velocity component in null space. An advantage to this approach is the lack of the "removable" singular points associated with the RR technique. The NV technique uses the same basic information used in the RR technique, rather than ϕ and $\frac{\partial l}{\partial \theta}$. The computational cost of integrating a particular solution from specified initial conditions using the NV formulation requires an amount of computation that is at least comparable to the pseudo-inverse and Extended Jacobian techniques.

The disadvantage of the NV technique, relative to the RR technique, is the parameter α has little to do with the configuration of the manipulator at any given time. Its first derivative, $\dot{\alpha}$, is related to the nullspace velocity. By implication, one might assume α is related to the nullspace velocity. By implication, one might assume α is related to some distance traveled in the nullspace direction, but this is a path dependent integral, so α need not necessarily take on the same value for the same manipulator configuration if the trajectories are not identical. One available option is to integrate a subsidiary equation, such as $\dot{\phi} = m^T\dot{\theta}$, rather then integrating $\dot{\alpha}$ to obtain α, since α is not required in the formulation. This would provide a history of the self-motion of the manipulator over the trajectory.

4.3 Boundary Value Problems

With the computationally efficient methods for obtaining solutions to Problem (15) in hand, the next issue is that of obtaining particular solutions associated with given initial conditions or boundary values. The sections that follow will pose each type of boundary problem in turn, and provide a numerical method for obtaining solutions to the problem.

4.3.1 Initial Boundary Value Problem (IBVP)

The initial boundary value problem is the simplest problem. The initial orientation and velocity of the manipulator is specified by the user, subject to the kinematic constraints. It is useful to specify the initial joint angles with a redundancy resolution parameter or parameters to avoid imposing a requirement on the user to specify a full joint angle set consistent with the kinematic restraint. This allows the user to specify the workspace position and manipulator orientation in its self-motion at that position independently, rather than forcing the user to compute a joint angle set corresponding to the desired configuration. The initial position, then, is specified by the kinematic constraint in conjunction with a user-specified initial value of θ.

The "optimality" of the solutions generated by all the initial value techniques presented must be verified. This is a direct consequence of the fact the Euler-Lagrange equations from which they are derived are only necessary, but insufficient, conditions for optimality. A solution generated from an arbitrary set of initial conditions may well be a locally maximum cost solution, or may correspond to a solution that is first order stationary, but for which large changes in trajectory produce lower cost solutions.

4.3.2 Natural Boundary Value Problem

The "natural" boundary value problem occurs when there are essentially no conditions on the configuration of the manipulator at either endpoint, and we wish to find initial and final configurations that yield the least-cost solution. A necessary condition for the solution to the natural boundary value problem is the nullspace joint velocity be zero at the initial and final configurations.

The approach developed to solve this boundary value problem uses the solution to the IBVP. The NVBP solution, then, can be reduced to finding the zeros of a function that is computed by solving the IBVP. This approach provides solutions that satisfy the necessary conditions for optimum solutions to the NBVP, but to find the actual optimum all solutions must be examined.

The computational requirements imposed by the requirement to examine the entire range of solutions to the IBVP is obvious. The worst case computation cost of the solution can be immense. Rather than integrating the NV equations once, as was required for the IBVP, the NBVP requires, in principle, infinitely many such evaluations. However, many practical motion profiles give rise to a relatively smooth function for the nullspace velocity component $\dot{\alpha}^T$, and the zeros of this function can be isolated with a small number of evaluations of the IBVP.

A final aspect of this solution technique is poor performance, as might be expected, when the initial (or final) configuration is itself near a kinematic singular point.

4.3.3 Two Point Boundary Value Problem (TPBVP)

Solutions to the two point boundary value problem can be obtained by a method analogous to that used for the NBVP. In this problem, ϕ_0 and ϕ_T, or equivalent information is given. The solution to Equation (19) is required and can be found by making use of the IBVP solution. The TPBVP approach takes ϕ_0 as the configuration initial condition and searches for a velocity initial condition, $\dot{\phi}_0$, leading to a solution with ϕ_T as the final value of ϕ.

In general, it is likely that ϕ will completely resolve the redundancy. Specification of additional parameters should allow θ_T to be known unambiguously.

4.3.4 Periodic Boundary Value Problem (PVBP)

This is the problem of finding the least cost periodic motion for $\theta(t)$ corresponding to a workspace motion $x(t)$ that is also periodic, or cyclic. That is, we have a situation where $x(0)=x(T)$, and we wish to find $\theta(t)$ that is a solution to the problem of Equation (1), and meets the additional constraints $\theta(0)=\theta(T)$ and $\dot{\theta}(0)=\dot{\theta}(T)$. This results in a joint angle time history that follows the desired trajectory, is periodic, and is low cost in the sense of Equation 19.

This problem differs from the previous boundary problems as it requires a search in two variables, ϕ_0 and α_0, for the simultaneous zeros of two expressions that specify the problem. Intersections of plots of solutions to these expressions will correspond to solutions, but spurious solutions will have to be rejected.

4.4 Summary

This section has presented a new technique for generating globally optimal solutions (in a velocity-magnitude squared sense) to the inverse kinematics of redundant manipulators, due to Nakamura. The section discussed the computational requirements of the techniques and showed derivations of two reduced order methods. It presented solutions related four different types of boundary problems. The techniques presented are a practical off-line means of finding good solutions to the inverse kinematics of redundant manipulators.

5.0 Dynamics and Control

This section presents a local and a global optimization method for minimizing torque leading at the joints in the least-squares sense. The local optimization technique minimizes torque by specifying a null space vector using a generalized inverse applied to accelerations. The local method is compared to a straightforward pseudo-inverse and an inertial-weighted pseudo-inverse. The global optimization method is formulated through the use of calculus of variations, and is compared with the local algorithms.

5.1 Local Torque Optimization

Redundancy resolution using local torque optimization can reduce joint torque and avoid joint torque limits throughout the manipulator movement. An effective approach is to keep the joint torques close to the midpoint of their upper and lower torque limits. This is done in a least squares sense by minimizing a vector that combines a vector of the upper limits of the joint torques and the vector of lower limits. For simplicity, these limits are assumed motion independent. The idea of different available torque ranges is easily solved by using a weighting matrix with proper representation of the available torque ranges.

The algorithms we investigated are:

- Unweighted pseudo-inverse algorithm (UPI)

$$\underline{\tau} = HJ\dagger(\ddot{x} - \dot{J}\underline{\dot{\theta}}) + \underline{C} + \underline{g} \tag{31}$$

- Inertia-weighted pseudo-inverse algorithm (IWPI)

$$\underline{\tau} = HJ^{\dagger}_{H}(\dot{x} - \dot{J}\underline{\dot{\theta}}) + \underline{C} + \underline{g} \tag{32}$$

- Unweighted null-space algorithm (UNS)

$$\underline{\tau} = HJ\dagger\left(\underline{\dot{x}} - \dot{J}\underline{\dot{\theta}}\right) + \underline{C} + \underline{g} + H\,[\,H(I - J\dagger J)\,]\,\dagger\frac{\left(\underline{\tau}^{+} + \underline{\tau}^{-}\right)}{2} \tag{33}$$

- Weighted null-space algorithm (WNS)

$$\tau = HJ\dagger(\,\underline{\alpha} - J\underline{\theta}\,) + \underline{C} + \underline{g} + H\,[\,W^{*}H(\,I - J^{*}J\,)\,]\,\dagger\,W^{*}(\underline{\tau}^{+} + \underline{\tau}^{-}\tfrac{)}{2} \tag{34}$$

The unweighted pseudo-inverse algorithm derives the joint torques without the null space component yielding a solution with minimum $\dot{\theta}^T\dot{\theta}$. Presumably, this should keep joints from moving too fast if started at rest, possibly yielding a more controllable motion. The inertia weighted pseudo-inverse algorithm [9, 10] yields a minimum kinetic energy solution. The unweighted and weighted null-space algorithms are the proposed methods presented in the previous section.

5.1.1 Results

Performances of the unweighted null-space (UNS), unweighted pseudo-inverse (UPI) and inertia-weighted pseudo-inverse (IWPI) algorithms were compared for representative trajectories and assumed characteristics of a basic three-link planar rotary manipulator.

For a short movement, the UNS dramatically reduces the joint torques over the UPI, with the IWPI falling somewhere between the two. A dramatic reduction in joint torque of the UNS is the main contribution to the overall increase in performance. For a medium length movement, the UNS still shows a dramatic reduction over the USI, with the IWPI again falling in between.

The situation changes considerably for a long movement. Both the UNS the IWPI algorithms show unexpected instability near the end of the movement. The instability seems to be caused by the alignment of the second and third links and the large joint velocities associated at the time of alignment. The redundancy of the arm is partially lost in the first joint at the alignment, and the large joint velocities require extremely large joint torques to keep the manipulator on the desired trajectory. Evidently, the UNS and IWPI algorithms always show instability for relatively long trajectories.

The UPI algorithm appears to be more stable. There were a few trajectories where only the WPI showed the instability. The UPI algorithm goes through a partial loss of redundancy in the third joint near the movement midpoint, and another loss of redundancy in the second joint near the end of the movement. These losses of partial redundancy together with the large joint velocities seemed to have caused the instability of the UPI algorithm.

In the WNS for the same trajectories, the third joint torque is pulled much closer to its midpoint at the expense of the first and second joints. However, all the joint torques are well within their ranges. Unfortunately, the WNS also shows this instability in long movements. There were even movements where the instability is shown only by the WNS. The characteristics of the instability in the weighted case were identical to those of unweighted cases.